MULTIPLE REGRESSION ANALYSIS APPROACH TO AUTOMATIC DESIGN OF ADAPTIVE IMAGE PROCESSING SYSTEMS

N. Otsu, T. Kasvand, and S. Umeyama

ERB-996

NRCC No. 27428

FEBRUARY 1987

TABLE OF CONTENTS

Page

ABSTRACT v
RÉSUMÉ vi
1 INTRODUCTION 1
2 DEFINITIONS AND FORMULATIONS 2
3 MULTIPLE REGRESSION ANALYSIS 4
3.1 General Description 4
3.2 Multiple Regression Analysis Formulation for Space-invariant Processing 5
4 SOME EXAMPLES OF PRACTICAL APPLICATIONS 7
4.1 Adaptive Filter Design 7
4.2 Estimation of Blur and Restoration 8
4.3 "Blind" Restoration 10
5 CONCLUDING REMARKS 16
ACKNOWLEDGEMENTS 16
REFERENCES 16
APPENDIX A. MEMORANDUM OF HOW TO USE GIPSN 19
A1 INTRODUCTION 19
A2 OVERVIEW OF THE SYSTEM 19
A3 DETAILED EXPLANATION OF EACH BLOCK 20
A3.1 Terminal Identification 20
A3.2 Norpak Initialization 21
A3.3 Menue 22
A3.4 Read Image Data 24
A3.5 Image Data Normalization 25
A3.6 Colour Control 26
A3.7 Display Erasing. 27
A3.8 Display Images 27
A3.9 Replace Images 28
A3.10 Add Noise 28
A3.11 Thresholding 29
A3.12 Edge Detection 29
A3.13 Filterings 30
A3.14 MRAF 31
A3.15 KL-Filtering 34
A3.16 SVD 36
A3.17 END 36
A4 DOCUMENTATION OF SUBROUTINES 36
A4.1 NORPKM 36
A4.2 NORPK 38
A4.3 The Other Subroutines 39

TABLE OF CONTENTS (Continued)

	Page
REFERENCES	45
APPENDIX B. SOURCE LIST OF GIPSN.FOR	47

LIST OF FIGURES

1	Schematic diagrams	3
2	Examples of adaptive linear filter design	9
3	Image restoration under an unknown degrading transmission channel	11
4	Examples of image restoration by using the MRA approach	14
A1	General flow-chart of GIPSN	21
A2	Menue and related memories	23

LIST OF TABLES

A1	Arrays and parameters declared and used in GIPSN	22

ABSTRACT

From a conceptual point of view, all image processing operations which convert one image to another image may be looked upon as transformations of the form

input image → a process → desired output image

Similarily, pattern recognition operations for identifying objects in images may be viewed as

input image → a process → object class

The goal in both cases is to determine the processes which achieve the desired output images or objects classes. The usual approaches to these problems are based on the so-called trial and error methods where a method is proposed, tried on many images, and checked for results. This requires much experience on the part of the designer.

Multiple regression analysis for modeling the correspondence between a variate or a set of variates and another set of variates seems to be one of the most promising and direct approaches to automatic design of adaptive (or learning) systems for image processing and computer vision. The advantage of such an approach is the capability to produce in an automatic and general way the functional "black boxes" (solutions) which are imposed by real problems, regardless of the inner details of these "boxes."

Some approaches are shown with experimental results, such as automatic design of adaptive filters for image enhancement and restoration by giving the input image and the desired output image as a pair.

RÉSUMÉ

Théoriquement, toutes les opérations de traitement des images par lesquelles une image est transformée en une autre peuvent être considérées comme des transformations de la forme

image d'entrée → processus → image de sortie désirée

De même, les opérations de reconnaisssance des images pour identifier des objets dans des images peuvent être considérées comme suit

image d'entrée → processus → classe d'objets

Le but dans les deux cas est de déterminer les processus qui produisent les images de sortie désirées ou les classes d'objets. Les méthodes habituelles de traitement de ces problèmes procèdent par tâtonnement: une méthode est proposée, essayée avec plusieurs images, et des résultats sont vérifiés, ce qui exige beaucoup d'expérience de la part du concepteur.

L'analyse de régression multiple pour modéliser la correspondance entre une variable ou un ensemble de variables et un autre ensemble de variables semble l'une des méthodes les plus prometteuses et les plus directes pour la conception automatique de systèmes adaptifs (ou d'apprentissage) en matière de traitement des images ou de vision par ordinateur. L'avantage d'une telle méthode est de pouvoir produire d'une manière automatique et générale les "boîtes noires" (solutions) fonctionnelles qu'imposent les problèmes réels, sans égard aux détails internes de ces "boîtes".

Quelques méthodes, avec résultats expérimentaux, sont décrites : conception automatique de filtres adaptifs pour l'amélioration des images et remise en état avec spécification d'une paire image d'entrée-image de sortie désirée.

MULTIPLE REGRESSION ANALYSIS APPROACH TO AUTOMATIC DESIGN OF ADAPTIVE IMAGE PROCESSING SYSTEMS

N. Otsu, T. Kasvand, and S. Umeyama

1. INTRODUCTION

Image processing and computer vision are current main topics for machine realization of visual information processing systems the performance of which would resemble human visual abilities. There are various problems to be solved (1) and many methods and systems have been proposed so far. Most of those methods are aimed at individual problems and the solutions are based on intuitive and ad hoc techniques. Trial and error methods are used to evaluate the proposed methods. Thus, the evaluation is left to the somewhat subjective and uncertain human inspection and judgement process (feedback loop via human). It is usually not easy to design computer vision systems by integrating the many techniques that are available (a tool box of techniques). It requires much empirical knowhow, viz., which technique is applicable to which problem and in which case. The resultant systems tend to lack adaptibility, self-learning capability, and generality.

Conversely, if a system possesses a general and adaptive structure (frame) and a criterion for quantitative evaluation of results, this implies that such a system can design itself automatically and optimally for a purpose and can adapt itself to required changes. Many of the basic problems in the field of image processing and computer vision and also pattern recognition can be formulated in the unified frame of multiple regression analysis, for example, image processing (2), linear and nonlinear feature extraction in statistical pattern recognition (3), and associative memory (4). Such an approach seems to be one of the most direct, simplest, and promising approaches to automatically designing adaptive (or learning) systems for image processing and computer vision. The theory is briefly outlined and illustrated with examples.

2. DEFINITIONS AND FORMULATIONS

Let an image be represented by an m by n matrix G. The (i, j) element of G, a pixel, takes a scalar value of some characteristic, such as gray level, and is denoted by g(i, j) or g(k), or simply referred to as g if not confused, where k stands for a position index, reordering (i, j) in appropriate order. The m by n image matrix G = [g(i, j)] can be regarded as an mn-dimensional vector (an mn by 1 matrix) $\underline{G}$ = [g(k)]. The characteristic value g is generally assumed to be real, so that we shall consider that images form a mn-dimensional vector space with a metric in the sense of vector norm.

Denote a local neighbourhood at k by N(k), and let a set of characteristics measured on N(k) be denoted by a vector $\underline{g}(k)$ or simply $\underline{g}$. Such a vector will hereafter be called a local neighbour characteristic vector and be referred to as LNCV. A simple example of LNCV is a vector which consists of the pixel values g's themselves within N(k). A more general example may be a feature vector extracted from N(k).

A feature vector extracted from an entire image G is denoted by $\underline{x}(G)$. A nominal category name (e.g., class name) and a scalar value (e.g., measurement) assigned to an image G are denoted by c(G) and y(G), respectively.

Image processing is generally formulated by a mapping t from an input image G to a desired output image H, i.e., H = t(G), see Fig. 1a. In practice, H is only conceived as an ideal output, and the mapping t is implicit and unknown. Consequently, we usually propose some definite processing s which is supposed to approximate t and obtain an output image F = s(G), expecting F to be as close to H as possible. When the processing is supposed to be space invariant (or shift invariant), then the mapping t can be reduced to one from LNCV $\underline{g}(k)$ of G to a pixel value h(k) of H, viz., $h = t(\underline{g})$ (a filtering). Furthermore, if the processing is linear, then the mapping t can be represented by a convolution and simply given by an inner product between LNCV $\underline{g}$ and a weight vector $\underline{w}$, $h = \underline{w}'\underline{g}$, where the symbol ' represents the transpose.

In computer vision or image measurement and image recognition, a system is expected to realize the mapping t from a given image G to a measurement y(G) or a category name c(G) which should be assigned to the image G. Usually, such a process is treated in two stages. The first stage is to extract a feature vector $\underline{x}(G)$ from G (feature extraction), and the second stage is to assign y or c, based on the feature vector $\underline{x}$ (estimation or decision making).

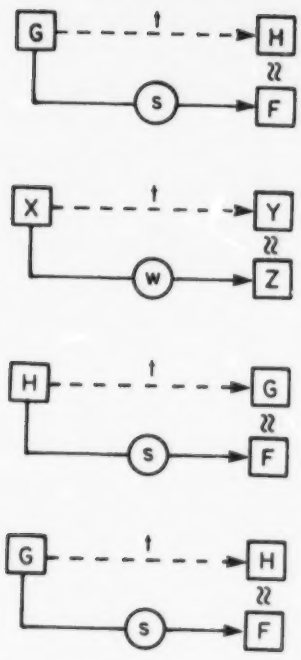


Figure 1. Schematic diagrams. G and X, given input image and variates; H and Y, ideal (desired) image and variate(s); F and Z, actually obtained output image and variate(s); t, ideal implicit mapping (processing); s and w, actual explicit mapping (processing); F and Z are expected to be as close to H and Y as possible. (a) A model of image processing. The input image G is given and an ideal output image H is desired, but the necessary mapping t is unknown. Based on experience, a mapping s is tried, which generates an approximation F of H. (b) Multiple regression analysis. A variate X is given and the variate Y is desired. The mapping t is unknown. A mapping w is computed which generates an output Z which approximates Y in terms of least squares. (c) Estimation of the blurring process, where H is the unblurred image and G is the given blurred image. The unknown blurring process t is estimated by model s which minimizes the least mean square error between G and F. (d) The restoration process is essentially the inverse of (c), where the unknown restoration mapping t is modelled by s, producing the restored image F. H is the idealized unblurred image and G is the given blurred image.

3. MULTIPLE REGRESSION ANALYSIS

As has been seen in the previous section, many of the problems in the field of image processing and computer vision can be formulated as problems of how to realize the correspondence between a set of input variates and an output variate (or another set of output variates). Such variates are usually given by real values and sometimes given by nominal variates (category or class names). This kind of problem is well known in multivariate data analysis as the multiple regression analysis (5).

3.1 General Description

The multiple regression analysis (hereafter abbreviated as MRA) tries to simulate an unknown functional black box t, or correspondence between input variates (say a vector $\underline{x}$) and output variates (say a vector $\underline{y}$), in terms of the following linear regression model

$$[1] \qquad \underline{z} = W'\underline{x}$$

where W is a matrix, see Fig. 1b. The criterion measure to evaluate the model fitness is the least-mean-square error, namely to minimize the following mean squares with respect to W.

$$[2] \qquad e(W) = E\|\underline{y} - \underline{z}\|^2$$

$$= E\|\underline{y} - W'\underline{x}\|^2$$

$$= \mathrm{tr}[W'E(\underline{x}\underline{x}')W - 2W'E(\underline{x}\underline{y}') + E(\underline{y}\underline{y}')]$$

where E denotes an operator for taking expectation (averaging) with respect to the argument which is implicit ($\underline{x}$ in this case), $\|.\|$ denotes the squared norm of a vector, and tr is the trace of a matrix. By letting the differentiation of e(W) with respect to W be zero (the normal equation), the optimal solution of W is given by the following.

$$[3] \qquad W = Rxx^{-1}\, Rxy$$

where $Rxx = E(\underline{x}\underline{x}')$ is an auto-correlation matrix of $\underline{x}$, $Rxy = E(\underline{x}\underline{y}')$ is the cross-correlation matrix between $\underline{x}$ and $\underline{y}$, and the power of -1 means the inversion of the matrix.

If the desired output is nominal and nonmetric, such as a set of category names in pattern recognition problems, then we can also treat them in the above metric frame by regarding them as orthonormal base vectors $\underline{y}$, allotting each dimension to each category. In such a case MRA takes the

form of the so-called least-mean-square discriminant analysis. It has been shown that the optimal nonlinear solution is closely related to the Bayesian decision and also related to the usual linear discriminant analysis in the linear case (3).

In the case of finite samples, e.g., k pairs of correspondence ($\underline{x}_i$, $\underline{y}_i$), let $X = (\underline{x}_1, \ldots, \underline{x}_k)$ and $Y = (\underline{y}_1, \ldots, \underline{y}_k)$ be matrices, then the optimal W in Eq. [3] is given by

$$[4] \qquad W = (X^+)'Y' = (YX^+)'$$

where X^+ is the pseudo-inverse (or generalized inverse) of X and is given by $X^+ = X'(XX')^{-1} = X'(kRxx)^{-1}$ if the Rxx is nonsingular (6). The matrix W can be computed recursively with respect to k, which enables the MRA model to be equivalent to an adaptive process. Such an MRA model has been applied as a model of associative memory (4).

It should be remarked that the linear formulation can also include a class of nonlinearities because the input vector $\underline{x}$ is open to definition, so that we have only to redefine $\underline{x}$ by any nonlinear terms of the former terms.

If the unknown function (mapping) t can be represented by the MRA model, in other words, t is included in the set of mappings S which can be described by the MRA model, then t can be exactly represented by the optimal solution $s \in S$ with the result that the least-mean-square error (LMSE) becomes zero. Conversely, when t cannot be represented by the MRA model, which is the usual case, i.e., t is not included in S, then the optimal solution obtained is the least-mean-square approximation of t in terms of S (a projection of t on S). Now the value of least-mean-square error shows the degree of fit to the MRA model. Of course, the more complicated and powerful MRA model we make by introducing additional parameters for nonlinear terms, the more will LMSE be reduced. However, this is an endless problem, and we need some tradeoffs in practical applications. For the model selection problem several criterion measures have been proposed (see, for example, ref. 7).

3.2 Multiple Regression Analysis Formulation for Space-invariant Processing

In many cases of filter design, the filter can be considered local and space-invariant. Then the mapping can be performed by a local operation within some effective neighbourhood N(k) over G (e.g., inverse filtering).

Consequently, the operation is formulated as a mapping s from LNCV $\underline{g}(k)$ of the input image G to the pixel value f(k) of the output image F. In the linear case s is given by a convolution, and the weighting matrix W in Eq. [1] is reduced to a vector $\underline{w}$. The principle is as follows. Once LNCV is defined, MRA model Eq. [1] determines a model of filters (a class S of s) which are given by the following.

$$[5] \qquad f(k) = \underline{w}'\underline{g}(k), \quad \underline{F} = [f(k)], \quad \underline{G} = [g(k)]$$

The mean-square error Eq. [2] is calculated between the filter output f(k) of F and h(k) of the desired output image H over G, and MRA solution Eq. [3] yields the optimal filter within the class S.

$$[6] \qquad w = Rgg^{-1}\,\underline{r}$$

where

$$Rgg = E[\underline{g}(k)\underline{g}(k)']$$

and

$$\underline{r} = E[h(k)\underline{g}(k)]$$

A form of control over the MRA process is obtained by introducing a learning weight matrix (mask) P(k), such that

$$[7] \qquad P(k) > 0$$

$$\sum_k P(k) = 1$$

where the averaging (expectation) is performed as follows

$$[8] \qquad E(*) = \sum_k P(k)(*)$$

The present method is stable in the presence of additive noise, which is inevitably associated with blurred images in particular, since the noise in the input image G is counterbalanced by the MSE (mean square error). Also, better results may be expected if the model S, namely LNCV, is changed to a more sophisticated (nonlinear) one.

4. SOME EXAMPLES OF PRACTICAL APPLICATIONS

The multiple regression analysis approach outlined above is a very powerful and formal approach to designing filters for image processing operations and also for object recognition. In words, the basic process may be characterized as follows. For filtering operations, two images are required, namely the input image G and the desired output image H, see Eq. [1], where the variates $\underline{x}$ and $\underline{z}$ represent G and H. If a sufficient number of pairs of input and output images are given, the MRA process can yield the matrix W which is the best filter to convert G to H, see Eq. [3]. Basically, no image processing knowhow is required by the user, since the process is totally automatic.

In the case of space-invariant filters the calculations can be considerably reduced by selecting local neighbourhood characteristic vectors (LNCV's), from which the mapping is computed, see Eq. [6]. Thus the operator only needs to define the local area size of the filter.

In certain situations, as in "blind" restoration, where the ideal output image is unknown, the user has to "bias" the MRA process in the desired direction by using a weight mask or matrix P, which is an image which indicates the importance to be attached to the various pixels.

Almost all the examples in this section were generated by using GIPSN (General Image Processing System on NORPAK) which was implemented by one of the authors. Appendix A gives the memorandom of how to use GIPSN.

4.1 Adaptive Filter Design

Filtering is one of the most important techniques of image processing. As has been mentioned in the preceding section, it is formulated as a mapping t from LNCV $\underline{g}(k)$ of input image G to pixel value h(k) of output image H, $h = t(\underline{g})$, where H is a desired output from G and t is implicit and unknown. In usual approaches an explicit mapping (operation) s is taken by intuition to realize (approximate) t (e.g., differential operators for edge detection), and the output $F = s(G)$ is checked. If the result F is inadequate, another s is tried. Such an approach is not only tedious but also limited to the well-defined problems for which we can imagine some s for t (or how to do it).

The MRA approach to this problem can serve a quite automatic way of designing filters. (A similar idea has also been presented in Eq. [2], although the concept of MRA is not explicit there while the iterative solution is employed.) The principle is as follows. Once LNCV is defined, the MRA model shown in Eq. [1] determines a model of filters (a class S of s)

which are given by Eq. [5], where the output is a scalar in this case and W is a vector $\underline{w}$. The mean square error [2] is calculated between the filter output f(k) of F and h(k) of the desired output image H over G, and the MRA solution Eq. [3] yields the optimal filter within the class S, see Eq. [6].

It should be noted that such a solution depends on how LNCV is defined. However, the important point is that the solution is not explicitly defined but automatically derived from the MRA process. This means that a system based on MRA can automatically and adaptively design any filtering function which is shown by a pair (or pairs) of input image G and desired output image H (implying learning ability or trainability).

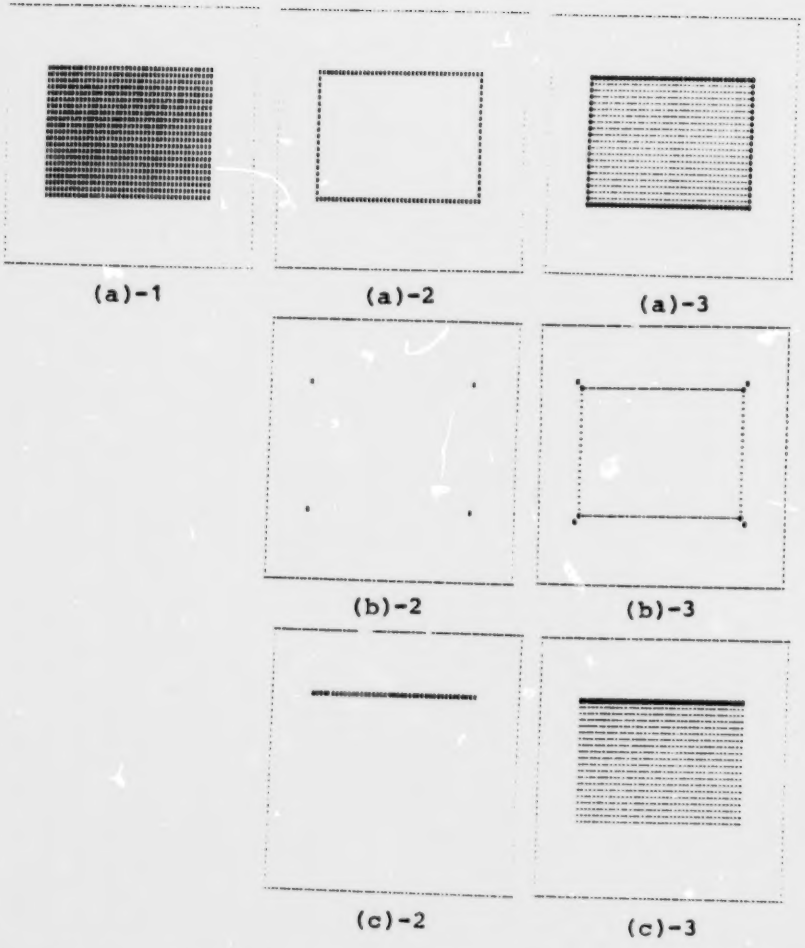
In Fig. 2, some examples of the experimental results are shown in the simplest case of linear filters, namely LNCV is confined to the vector consisting of gray levels of pixels within a local square mask over gray-level image G. Figure 2 a, b, and c are the cases of edge detection, corner detection, and only upper edge detection, respectively. The obtained weight vectors $\underline{w}$ in the case of 3 by 3 masks imply the following filters, respectively.

$$[6] \qquad \underline{w} = \begin{matrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{matrix}, \quad \begin{matrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{matrix}, \quad \begin{matrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

4.2 Estimation of Blur and Restoration

Image restoration is one of the most mathematically formulated topics in image processing (with the possible exception of image reconstruction). It deals with the problem of how to get a clear image from a given degraded (blurred) image. Many methods have been proposed so far (8-13). In most cases it is assumed that the original image is unknown but the blurring process is mathematically known or estimated. Therefore, image restoration is formulated as a problem of inversing the blurring process; namely, solving an algebraic equation (usually linear but quite high dimensional) or solving for an inverse filter (in case the blurring process is supposed to be space-invariant and representable by filtering).

However, there are several problems we encounter. First, such a mathematical inversion is generally very unstable, caused by the existing additive noise in the blurred image (10). To stabilize the restored image, some methods employ additional intuitive constraints on the resultant restored image, such as "smoothness" (10) or "sharpness" (11). Such an evaluation is, however, debatable. The second problem arises in the estimation of the blurring process, or its point spread function (PSF). Usually a typical PSF, such as a Gaussian, is assumed, and estimation of its



-0.463	0.000	-0.463
0.000	2.000	0.000
-0.463	0.000	-0.463

(a)-4

0.463	-0.950	0.463
-0.975	2.000	-0.975
0.463	-0.950	0.463

(b)-4

0.000	-0.975	0.000
0.000	1.000	0.000
0.000	0.025	0.000

(c)-4

Figure 2. Examples of adaptive linear filter design. (a) edge detection: (a)-1 input binary image; (a)-2 desired output image; (a)-3 output image obtained with 3 × 3 mask using MRA, given images (a)-1 and (a)-2; (a)-4 3 × 3 mask (weight vector) designed. (b) corner detection for (a)-1: (b)-2 desired output; (b)-3 output obtained with 3 × 3 mask; (b)-4 3 × 3 mask designed. (c) upper edge detection only for (a)-1: (c)-2 desired output; (c)-3 output obtained with 3 × 3 mask; (c)-4 3 × 3 mask designed.

parameters is performed. In actual situations, however, it is difficult to specify and estimate the PSF, specially if it is completely unknown. Therefore, an inaccurate modeling of PSF in such a case tends to yield a poor result, no matter how mathematically rigorous a treatment may be followed to obtain the inverse filtering. The final common problem is that restoration needs much computation time.

Image restoration is a problem of inverse filtering, namely, a given image G which is a blurred (filtered) image of an unknown image H, the problem is to realize an inverse filter to obtain a restored image F. Usually the blurring process (filtering) from H to G is assumed to be known or estimable, and the inverse filter is found mathematically and applied to G, as outlined above. In this section we shall consider another situation in image restoration, where the blurring process is unknown, but the original image H is known or available to check the unknown blurring process (14). An example is the problem of image restoration (calibration) under unknown degrading transmission channel, where a test image H is available. The filter design (or learning of filters) described in the earlier sections can be applied directly to the problem of estimating the unknown blurring process (filter) and also to the problem of image restoration (inverse filtering). In the former, we apply MRA in the direction from the test image H to the blurred image G, see Fig. 1c, and conversely in the latter, from G to H without solving for the inverse filter, see Fig. 1d. Some experimental results for a real image are shown in Fig. 3 of restoration under an unknown degrading transmission channel, where the ideal unblurred image is available as a test image. It is seen that the method is quite stable even for the considerable amount of additive noise (the S/N ratio is 45/10 measured in terms of standard deviations).

4.3 "Blind" Restoration

In realistic (practical) situations only the blurred image G is available and the blurring process is completely unknown. This is typical, for example, in accident situations where there may have been no time to adjust the camera accurately and/or the camera was shaken during exposure. Other souces of blurred images are from surveillance, where "something of interest" may have been photographed, but since the camera could not be focussed on "everything at once" the probability that the "region of interest" in the image is blurred is rather high. Thus, whatever the source of the blurred image, only the blurred or degraded image G is available, which may have been further degraded by a poorly adjusted TV camera.

In most situations only the blurred image G is available which may have been further degraded by numerous processes. Most of the degradations can be removed to a certain degree by suitable image processing (15).

(a)

1	2	1
2	3	2
1	2	1

(b)

1. 001	1. 999	0. 999
1. 998	3. 003	2. 001
0. 999	2. 002	0. 998

(c)

0. 001	-0. 002	-0. 004	0. 006	-0. 0
-0. 003	1. 006	1. 998	1. 001	-0. 0
0. 000	1. 997	3. 003	2. 001	0. 0
-0. 000	1. 000	2. 003	1. 000	-0. 0
-0. 003	0. 007	-0. 007	0. 001	0. 0

(d)

(e)

(f)

Figure 3. Image restoration under an unknown degrading transmission channel. (a) Original (or test) image (128 × 128, 256 gray-levels). (b) Blurring filter (assumed to be unknown). (c) 3 × 3 Blurring filter estimated by applying MRA from (a) to (e). (d) 5 × 5 Blurring filter estimated by applying MRA from (a) to (e). (e) Image (a) blurred by (b). (f) Image restored by 3 × 3 filter which was learnt by MRA from (e) to (a). The restoring filter is

0. 019	-0. 048	0. 019
-0. 051	0. 186	-0. 050
0. 022	-0. 053	0. 024

(g)

(h)

(i)

(j)

(g) Image restored by 5 × 5 filter which was learnt by MRA from (e) to (a). The restoring filter is

-0.013	0.043	-0.059	0.048	-0.016
0.041	-0.104	0.093	-0.107	0.042
-0.053	0.087	0.060	0.086	-0.049
0.054	-0.127	0.116	-0.122	0.044
-0.027	0.075	-0.095	0.071	-0.022

(h) Blurred and noisy image. Independent Gaussian noise with zero-mean and standard deviation $\sigma = 10$ was added to (e). (Cf. std. of the blurred image is $\sigma = 45$ in (e)). (I) Image

Consequently, at present we only concentrate on the removal of the blur. By using our knowledge and understanding of the contents of the degraded image, we can in most cases identify regions in G for which approximated versions of the ideal image H can be formulated. Of course, if the image G is so badly degraded that we cannot understand anything in it, then it is highly likely that the image is beyond recovery.

The ideal image is expected to contain sharp edges, point-like features, markings on objects, and so on. Given that we ourselves can identify such features in the blurred image, the problem is essentially how to introduce such knowledge into the MRA process (i.e., heuristic ways of introducing prior knowledge). Naturally, in a given practical application the introduction of such knowledge may be immediately obvious. In general, however, the following somewhat subjective methods may be tried, based on various assumptions.

(a) There is a subregion in the blurred image, representing a highly contrasting object with sharp boundaries. By extracting a region around this object in G and thresholding (16), a relatively acceptable idealized version of it may result. Consequently the restoring filter can be estimated from this subregion and applied to the whole image.

(b) The proposed method in (a) is subject to noise in the image. The "ideal" image obtained by thresholding can be "improved" by using various image processing techniques applicable to binary images, such as smoothing, dilatation, erosion, and contour detection. Prior filtering of the blurred image, for example with a median filter, may be helpful in reducing the noise content before thresholding.

(c) The edges in the blurred image, possibly after median filtering, may be computed (17) by choosing a local neighbourhood size commensurated with the degree of blur in the image. If at least most of these edges were sharp in the ideal image, the magnitude of the edge image values may be used as a learning weight P(k) in the MRA method. Thresholding of the

---

restored by 3 × 3 filter which was learnt by MRA from (h) to (a). The learnt filter is

0. 003	-0. 017	0. 003
-0. 020	0. 126	-0. 020
0. 005	-0. 022	0. 009

(j) Image restored by 5 × 5 filter which was learnt by MRA from (h) to (a). The learnt filter is

0. 002	0. 012	-0. 025	0. 015	-0. 001
0. 010	-0. 049	0. 041	-0. 052	0. 014
-0. 025	0. 043	0. 091	0. 042	-0. 022
0. 017	-0. 050	0. 036	-0. 046	0. 010
-0. 002	0. 013	-0. 024	0. 014	0. 000

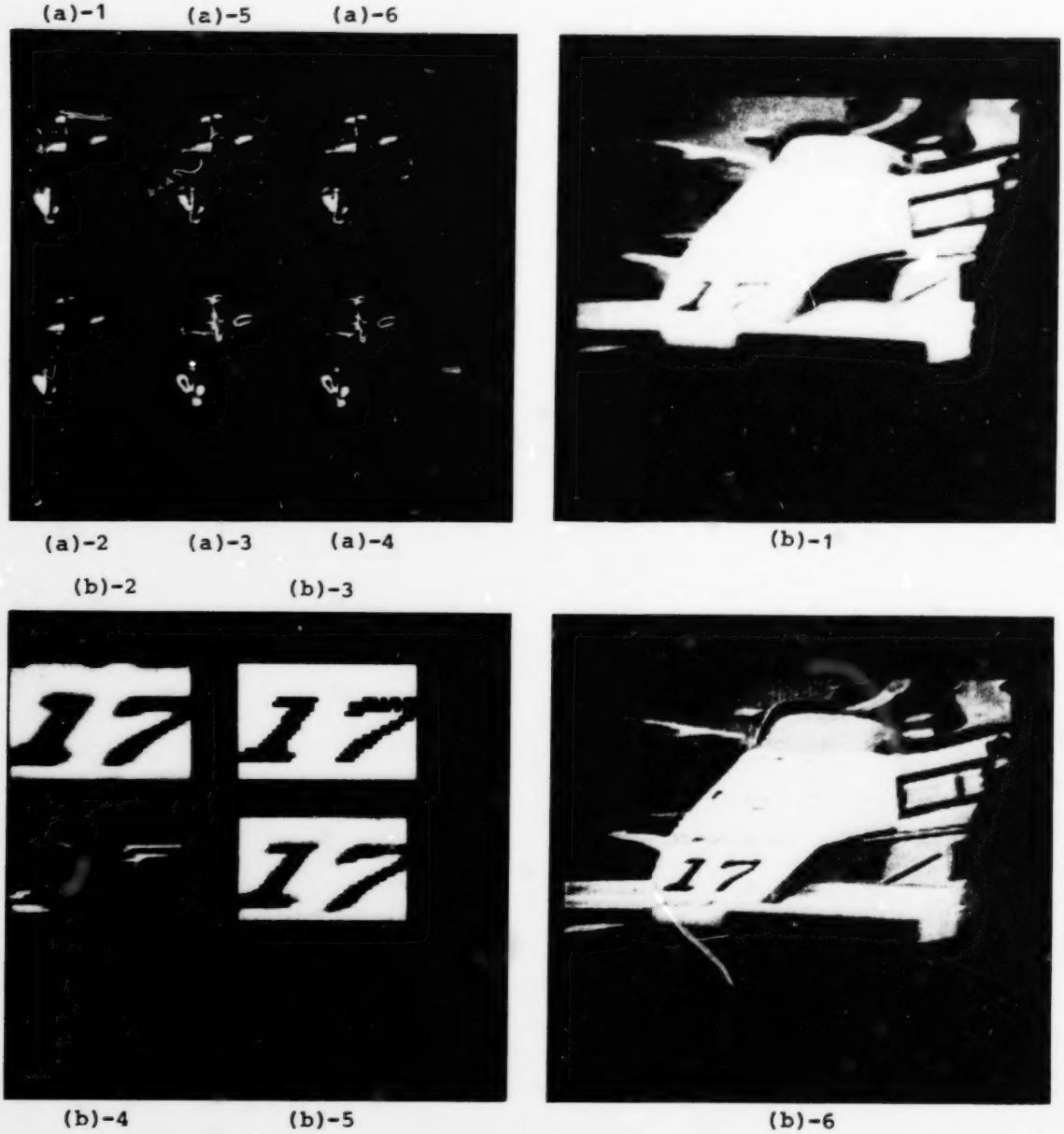


Fig. 4. Examples of image restoration by using the MRA approach. Our knowledge is introduced by identifying aspects of the image where the blurred information can be used to create "idealized" (sub)images. The learning of the restoration filters is based on approximating in the least squares sense these "idealized" images from the given blurred image. (a)-1 the given blurred image; (a)-2 binary images as ideal versions of the blurred image; (a)-3 learning weight, edge image computed from (a)-1; (a)-4 learning weight, thresholded image of (a)-3; (a)-5 the restored image based on the learnt restoration filters obtained from the MRA

edge image, followed by binary image processing methods, may also be used to produce a learning mask P(k) for selecting pixels from blurred and idealized versions of the images, in order to influence the MRA process in the desired direction.

(d) The process may be carried further, by actually attempting to create via on-line interaction idealized images of certain regions in the blurred image, which are then used in the MRA process.

Obviously, there are many ways of introducing our own knowledge about the world into the MRA process, thereby assisting the mathematical procedures in order to generate as good a restored image as our total knowledge permits. Examples of restoration with learning of the inverse filters are shown in Fig. 4. The given blurred image(a) is considered to be subjected to complex space-variant blurring consisting of shift and also rotation. It is shown that considerable local improvements can be achieved by using linear LNCV and learning processes.

---

process using the images (a)-1, (a)-2, and (a)-3; (a)-6 the restored image based on the learnt restoration filters obtained from the MRA process using the images (a)-1, (a)-2 and (a)-4; (b)-1 the given blurred image - in this case the blurring is likely to be highly space-variant; (b)-2 learning regions for estimation of inverse filters; (b)-3 binary images of (b)-2 as ideal versions of the blurred image; (b)-4 learning weight, edge image computed from (b)-2; (b)-5 restored regions; (b)-6 the restored image based on the learnt restoration filters obtained from the MRA process using the images (b)-2, (b)-3 and (b)-4 applied to the full image (b)-1.

5. CONCLUDING REMARKS

It has been shown that various basic problems in the field of image processing and computer vision can be formulated and solved in the unified frame of multiple regression analysis MRA. It suggests that a system based on an MRA model can be useful as a model of a general-purpose adaptive system for image processing and computer vision. The multiple regression analysis approach to image filtering and restoration under unknown degrading processes has been described. The method generates directly and automatically the restoring (inverse) filters in the learning process, given that an approximation of the unblurred image (or even a sub-region) can be constructed from the blurred one. The merits of this method are (a) self-learning ability for any inverse filter, (b) general and adaptable framework, (c) easy nonlinear extension, (d) stable and unique solution without ill-conditioning problems by virture of the LMSE criterion.

It should be noted, of course, that the result of restoration depends on the learning process. For example, the effect of edge sharpening vs. noise smoothing depends on the selection of the leaning mask (training area) or learning weight, and a learning biased in directionality will result in an enhanced restoration in that direction. These properties, however, allow us to control the learning process in the desired direction by introducing our knowledge about the image.

ACKNOWLEDGEMENTS

The authors are most grateful to both the institutes which made this work possible. Thanks are also due to many employees of the Division of Electrical Engineering, the National Research Council of Canada, especially to Dr. N.N. Abdelmalek, Mrs. C. Merritt, and Mr. A. Binch, for their valuable discussions and help in the use of the computer facilities.

REFERENCES

1. T. Kasvand. Robot vision and visual inspection. Proc. Graphics Interface '83, Edmonton, Alta. May 9-13, 1983. pp. 85-95.

2. S. Kondo and M. Matsuo. A theory of design for nonlinear functions of information processing on patterns. Proc. 5th International Conference on Pattern Recognition, 1980. pp. 965-967.

3. N. Otsu. Optimal linear and nonlinear solutions for least-square discriminant feature extraction. Proc. 6th International Conference on Pattern Recognition, Munich. October 19-22, 1982. pp. 557-560.

4. T. Kohonen. Associative memory - a system-theoretical approach, Communication and Cybernetics, Vol. 17, Springer. 1978.

5. T.W. Anderson. An introduction to multivariate statistical analysis. John Wiley. 1958.

6. C.L. Lawson and R.J. Hanson. Solving least squares problems. Prentice-Hall. 1974.

7. H. Akaike. A new look at the statistical model identification, IEEE Trans. Acoust., Speech, Signal Process. ASSP-19(6): 716-723; 1974.

8. H.C. Andrews and B.R. Hunt. Digital image restoration. Prentice-Hall Signal Processing Series. 1977.

9. T.S. Huang (Editor). Picture Processing and Digital Filtering, Topics in Applied Physics. Vol. 6. Springer-Verlag. 1975.

10. D.L. Philips. A technique for the numerical solution of certain integral equations of the first kind. ACM J. 9: 84; 1962.

11. G. Backus and F. Gilbert. Uniqueness in the inversion of inaccurate gross Earth data. Philos. Trans. R. Soc. London A266: 123, 1970;

12. C.W. Helstrom. Image restoration by the method of least squares. J. Opt. Soc. Am. 57(3): 297; 1967.

13. N.N. Abdelmalek and T. Kasvand. Digital image restoration using quadratic programming. Appl. Opt. 19(19): 3407; 1980.

14. N. Otsu. Multiple regression analysis approach to the automatic design of adaptive image processing systems. Proc. 27th SPIE Inter national Technical Symposium, San Diego, CA. August 1983.

15. T.C. Rindfleisch, J.A. Dunne, H.J. Frieden, W.D. Stromberg, and Ruiz, R.M. Digital processing of the Mariner 6 and 7 pictures. J. Geophys. Res. 76(2): 394-417; 1971.

16. N. Otsu. A threshold selection method from gray-level histograms. IEEE Trans. Syst., Man Cybern. SMC-9 (1): 62; 1979.

17. N. Otsu and T. Kasvand. A new family of nonlinear edge detectors for noisy images. Proc. 27th SPIE International Technical Symposium, San Diego, CA. August 1983.

APPENDIX A. MEMORANDUM OF HOW TO USE GIPSN

A1. INTRODUCTION

GIPSN is a general system (FORTRAN program) implemented on the VAX 11/780 and NORPAK image display for convenient interactive image processing experiments. It was originally developed by the author for his own use during his stay at DEE (Sept.'82-Nov.'83) and is still a provisional version. The system consists of basic subroutines for image processing and is designed to be so simple that one can easily use it and also modify it for special tasks by imbedding one's own subroutines.

The main program GIPSN.FOR is stored in [OTSU.IP], and the necessary subroutines are all in the subroutine library [OTSU.MVAL]MVAL.OBL. Hence the user can copy GIPSN.FOR to his account, compile it, and has only to link it with MVAL.OBL and another library [BINCH.RGP3050.VAX]ZLIB3050.olb for manipulating the NORPK image display. (For the convenience of linkage, copy command procedure [OTSU.IP]LKGIPSN.COM.) The source programs of the subroutines are found in [OTSU.IP] or [OTSU.MVAL].

The subroutine library MVAL contains about 300 subroutines useful for various mathematical computations, e.g., vector-matrix and eigenvalue computations, various multivariate data analysis techniques, and graphic input/output subroutines using PLOT-10 and TEKTRONIX 4014 type graphic terminal.

Before running GIPSN, assignment of image data files should be done. A command procedure [OTSU.IP]GIPSN.COM is provided for convenience, so that it suffices to input @GIPSN. The entry of image data can be changed by slight modification of the command procedure and the relevant part (READ DATA) in the main program.

A2. OVERVIEW OF THE SYSTEM

The basic philosophy of GIPSN is to provide a handy system for experiments in image processing research. Therefore it is designed as simply as possible in order to allow flexibility in modification and up-scaling. At present, the system reads an input image from preassigned files, processes it in various ways, and outputs the result to the NORPAK colour display.

The general flow-chart is shown in Fig. A1. Basically there is one big "MENUE" stage for selecting several different job blocks. Any selected block returns to MENUE stage after execution. Each block consists of some subroutines necessary to perform the respective job. At each level all the selection commands (parameters) and the "inevitable" parameters are

requested in an interactive manner. Therefore the user has only to input a number in the selection menue or a number (integer) corresponding to the parameter requested.

Several job blocks are already built in, such as image reading from data files, edge-detection, Karhunen-Loeve expansion, SVD, restoration, enhancement, and image display via NORPAK. It is easy to imbed additional new blocks.

The arrays to store images are declared in main program as one-dimensional and real-type ones, contrary to the conventional way of two-dimensional and integer-type ones. The intention is to simplify the treatment of dimensions, allowing flexibility and adjustability of dimensions in subroutine calls, and to make the processes free and compatible with various numerical computations. The arrays and parameters declared and used are listed in Table A1.

To avoid procedual complexity and explosion of file space, no file management functions have been installed, such as for storing of resultant images as new files. As a matter of fact, keeping the results in the form of camera films is more convenient and practical than storing them as computer files. Some attempt to issue ID numbers for resultant images was done in GIPSN, but it is incomplete. Users are expected to neglect those leftovers or make it complete.

In Fig. A2 is shown a diagram which indicates how each selection [1-15] in MENUE relates to the memories (image arrays).

A3. DETAILED EXPLANATION OF EACH BLOCK

In what follows, each job block is explained in detail with citation of the question answering parts.

The question (or the menue for selection) prompted is shown between broken lines, and + indicates the position to input the answer (an integer). The title in a broken line shows the content of the job following.

A3.1 Terminal Identification

This initial identification of terminal type is necessary for the optional function of graphic output of histograms which utilizes PLOT-10 and graphic terminals (TEKTRONIX-4014, VGT-100, etc.). In the case of the usual non-graphic terminals (VT-100, etc.), those outputs are automatically skipped.

```
-------------------------------
1= (TEKTRONIX) graphic terminal
2= other terminals
+
-------------------------------
```

A3.2 NORPAK Initialization

For handling NORPAK display facilities, the initialization should be done first of all. It includes memory erasing, initialization of all parameters, and loading of calibration tables for colour display.

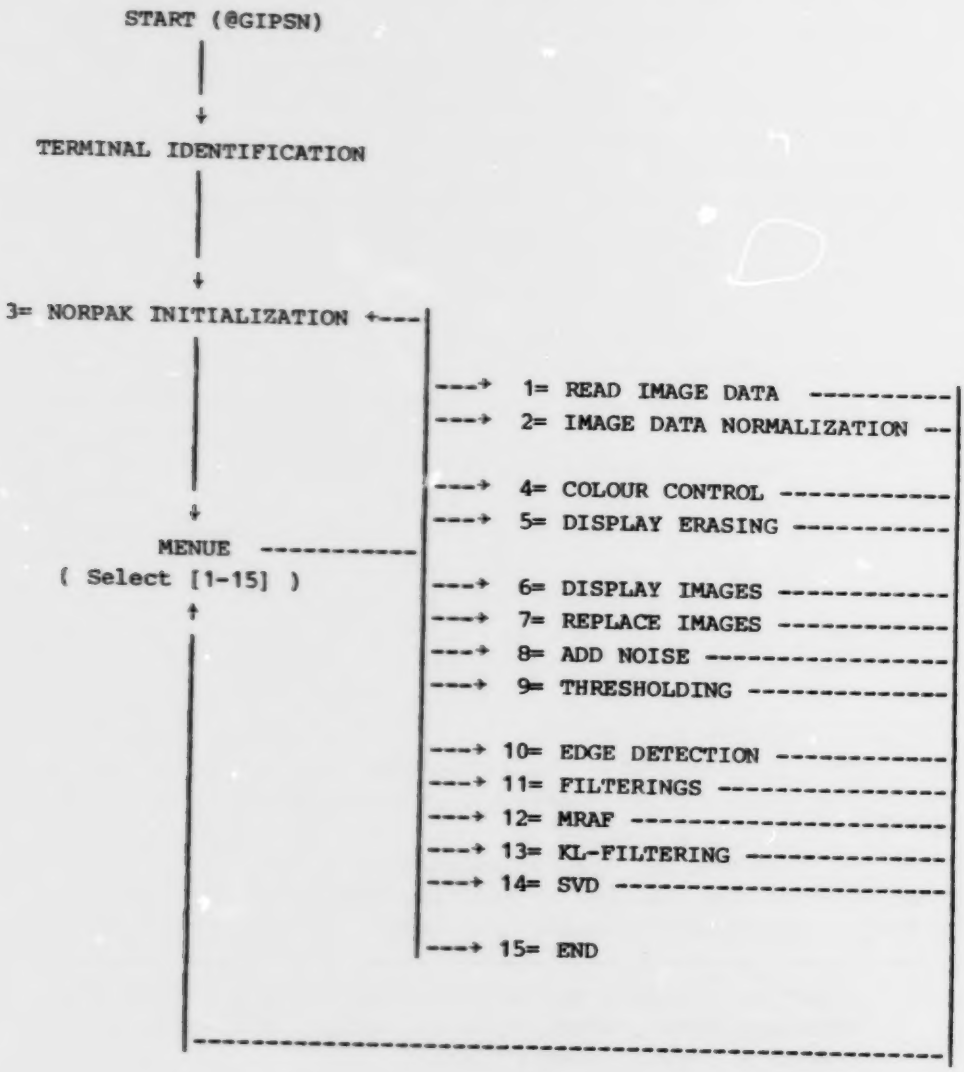


Fig. A1 General flow-chart of GIPSN

```
-------------------------
 INITIALIZATION OF NORPAK
 1=ERASE ALL OLD MEMORY
 2=NOT ERASE OLD MEMORY
 +
-------------------------
```

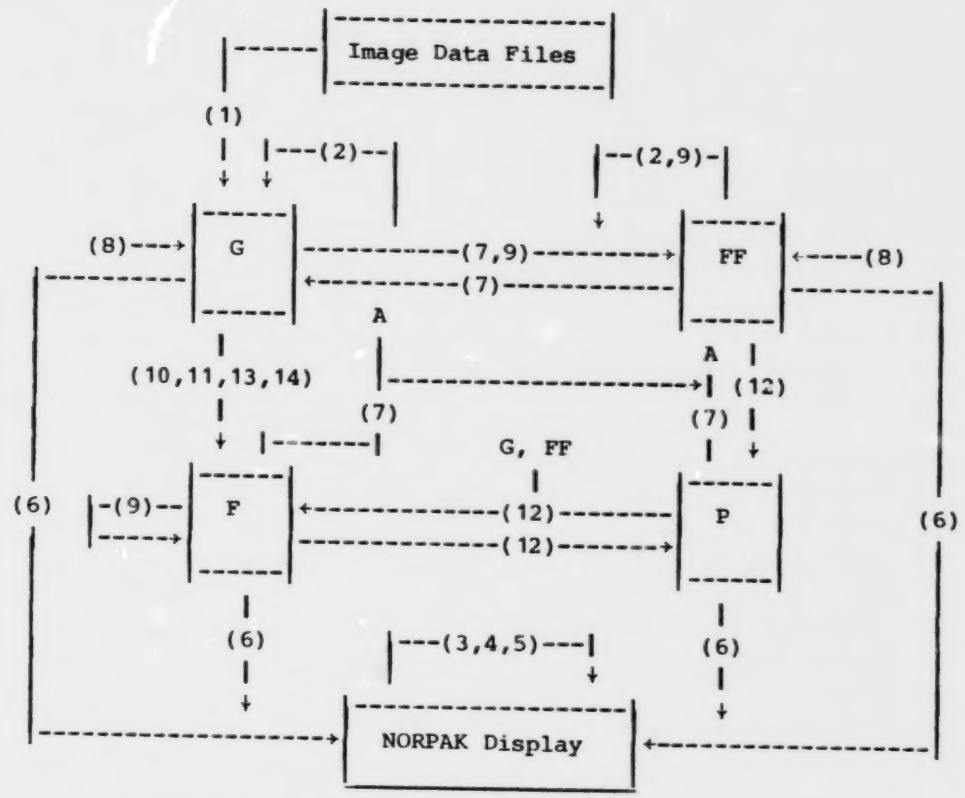
A3.3 Menue

This is essentially the initial stage of image processing. The first selection, therefore, should be Selection 1, viz., to read an image from

Table A1. Arrays and parameters declared and used in GIPSN

```
G(M,N) = input image data (M columns and N rows)
F(MM,NN) = output image processed
FF(M,N) = image buffer
P(M,N) = image buffer mainly for forced weighting mask
M = number of columns of images G, FF, and P
N = number of rows of images G, FF, and P
MM = number of columns of image F
NN = number of rows of image F
IJD = 400: maximum size of column or raw
MN = IJD*IJD: maximum number of total pixels of images

NL = 256: number of gray-levels of images
H(NL) = normalized histogram (unit sum)
ILTP, ILTI, TLTD, ILTT(NL) = look-up tables for colour display

ITERM = 1:TEKTRONIX (graphic terminal), 0:others
L = size (width) of local mask ( (2*L+1)**2  = LL )
LL = maximun dimension of local mask
LL2 = LL*LL maximum dimension of local statistics
R(LL*LL) = local statistics
EV(LL*LL) = eigenvectors for local statistics
W(LL), A(LL) = working areas, e.g. eigenvalues

IUN = unit number for reading image data
IPN = switch for positive or negative read of image
(NID = identification number of images)
```

```
      PARAMETER IJD=400, MN=IJD*IJD, NL=256, LL=100, LL2=LL*LL
      DIMENSION F(MN), G(MN), EV(LL2), W(LL), R(LL2), A(LL)
      DIMENSION FF(MN), P(MN), H(NL)
      INTEGER*2 ILTP(NL), ILTI(NL), ILTD(NL), ILTT(NL)
```



1= READ IMAGE DATA ..... Image Data Files ---→ G
2= IMAGE DATA NORMALIZATION ..... G --- G; FF ---→ FF
3= NORPAK INAITIALIZATION ..... NORPAK ---→ NORPAK
4= COLOUR CONTROL ..... NORPAK ---→ NORPAK
5= DISPLAY ERASING ..... NORPAK ---→ NORPAK
6= DISPLAY IMAGES ..... G,F,FF,P ---→ NORPAK
7= REPLACE IMAGES ..... F,FF --→ G; G,F,P --→ FF
8= ADD NOISE ..... Noise ---→ G,FF
9= THRESHOLDING ..... G --→ FF; F --→ F; FF --→ FF
10= EDGE DETECTION ..... G ---→ F
11= FILTERINGS ..... G ---→ F
12= MRAF ..... G,FF,P ---→ F; (FF, F--→P)
13= KL-FILTERING ..... G ---→ F
14= SVD ..... G ---→ F

Fig. A2 Menue and related memories

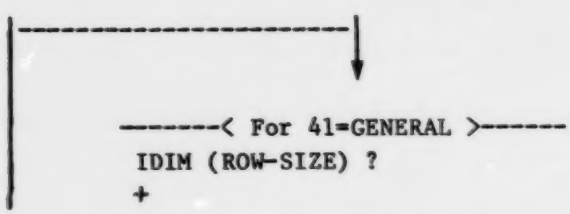
data files. In the menue prompted, the participating array(s), the processing flow, and also the jumping Label number in the program are indicated in ( ) for the help of understanding and easy cross reference.

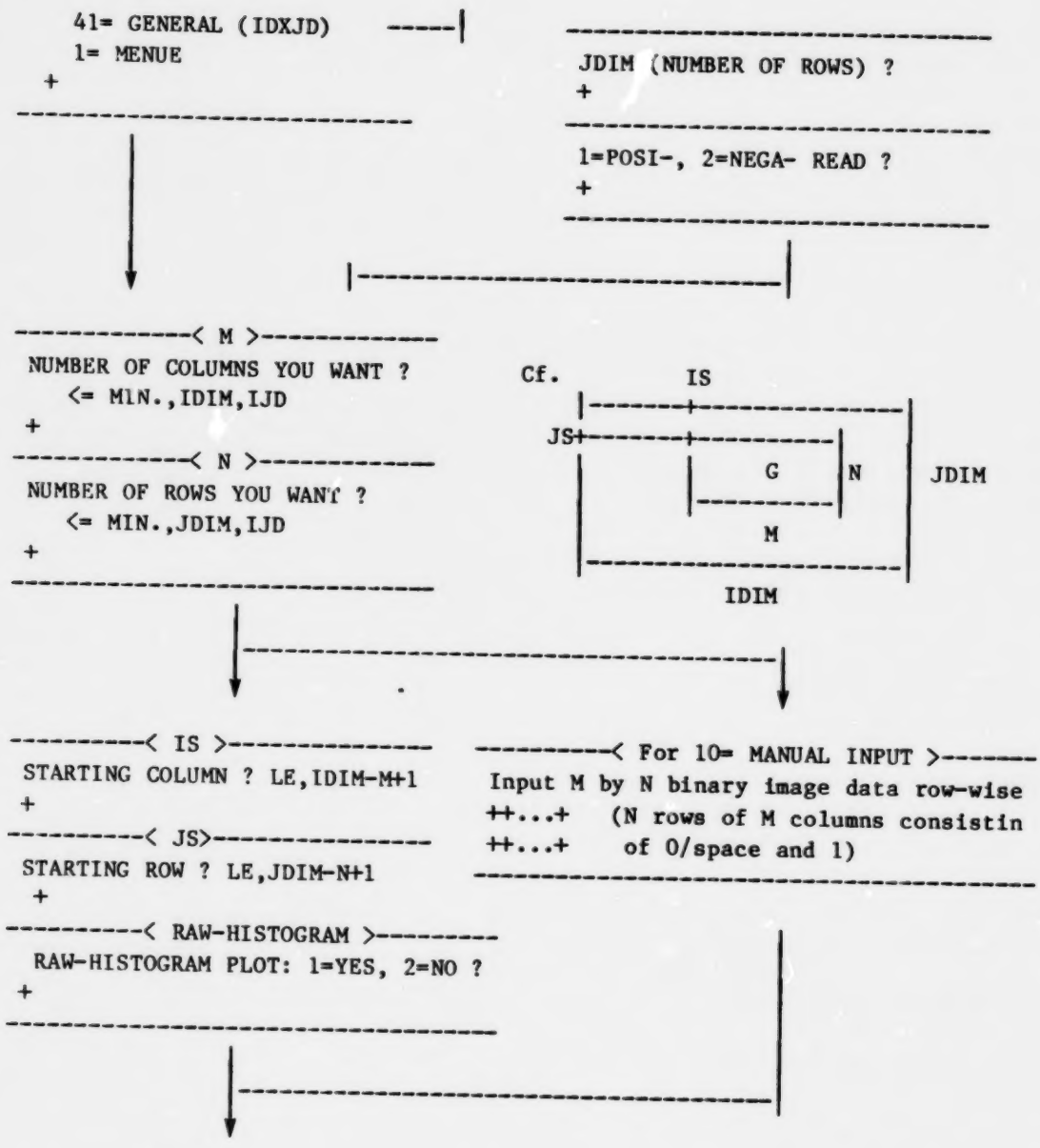
```
-----------------------------------------
   ========== MENUE ================
 1= READ IMAGE DATA (TO G, L1)
 2= IMAGE DATA NORMALIZATION (L21)

 3= NORPAK INITIALIZATION (L20)
 4= COLOUR CONTROL (L25)
 5= DISPLAY ERASING (L22)

 6= DISPLAY IMAGES (G,F,FF,P, L90)
 7= REPLACE IMAGES (G/FF WITH F/FF, L70)
 8= ADD NOISE (TO GFF, L6)
 9= THRESHOLDING (G TO FF / F / FF , L97)

10= EDGE DETECTION (G TO F, L4)
11= FILTERINGS (G TO F, L80)
12= MRAF (G&FF&P TO F, L5)
13= KL-FILTERING (G TO F, L3)
14= SVD (G TO F, L85)

15= END (L9)
   ==================================
SELECT MENUE
+
-----------------------------------------
```

A3.4 Read Image Data

In this block the user can select an image from preassigned data files and read in (cut out) any portion of it by inputting parameters required.

```
----------< IUN >------------
*** SELECT IMAGE DATA ***
    10= MANUAL INPUT
    11= GIRL (128X128)
    12= COUPLE (128X128)
    13= MOON (128X128)
    14= AIRPLANE (128X128)
    21= GIRL (256X256)
    31= CHROMS (640X512)
    32= FINGER (400X400)
```





A3.5 Image Data Normalization [S2, L21]

In some cases it is preferable to normalize an image data such that the mean and the variance of gray-levels are zero and unit, respectively. This block is optional for the purpose.

```
-----------------------------------------
 DATA NORMALIZATION (INTO MEAN=0 & SD=1)
 1= to G
 2= to FF
 3= EXIT TO MENUE
 +
-----------------------------------------
```

A3.6 Colour Control

This option is for colour control of the NORPAK display facility. The real-colour is to assign R, G, B to one of the bank 1, 2, 3, 4=nil, respectively. The gray-level is to assign R, G, B simultaneously to one of the bank 1, 2, 3. The pseudo-colour consists of two kinds of types, smooth and fine. The negation can be applied to the real-colour or the gray-level, and it is a flip-flop switch.

```
 ----< L25: COLOUR CONTROL >-------
  1= REAL-COLOUR (INITIAL DEFAULT)
  2= GRAY-LEVEL
  3= PSEUDO-COLOUR
  4= NEGATION (EXCEPT 3)
  5= EXIT (MENUE)
  +
 ---------------------------------

 --------< 1= REAL-COLOUR >--------

 COLOUR ASSIGN -
  R TO BANK [1,2,3] 4=OFF
 +
  G TO BANK [1,2,3] 4=OFF
 +
  B TO BANK [1,2,3] 4=OFF
 +
-------------------------------------
 1= TRY AGAIN, 2= EXIT ?
 +
-------------------------------------
          GOTO L25

--------< 2= GRAY-LEVEL >------------
 BANK ? [1,2,3]  4= EXIT
 +
-------------------------------------
```

```
          GOTO L25

-------< 3= PSEUDO-COLOUR >----------
 CALIB.TABLE: 1= SMOOTH, 2= FINE ?
 +
-----------------------------------
 BANK ? [1,2,3]  4= EXIT
 +
-----------------------------------
          GOTO L25
```

A3.7 Display Erasing

This option is sometimes needed in order to erase wrongly located outputs and to get more nicely laid out display. This allows us to erase a whole display bank (1, 2, or 3) or any subregion of it.

```
--------< Erasing >--------
1= BANK1
2= BANK2
3= BANK3
4= AREA  +
--------< Confirmation >---
ARE YOU SURE?  1=YES, 2=NO
 +
-----------------------------------
      GOTO MENUE

-------< For 4= AREA >--------------
 *** AREA ERASING MODE
 IDIM (ALONG ROW)?
 +
 JDIM (ALONG COLUMN)?
 +
-----------------------------------
     CALL NORPK(P,ID,JD,0.0,0.0,0)
     GOTO MENUE
```

A3.8 Display Images

At the stage of MENUE, users can display (peek) the current images stored in G (ORIGINAL), F (RESULTANT), FF (BUFFER), P (MASK), and also W (FILTER WEIGHT).

```
----------------------
1= ORIGINAL G
2= RESULTANT F
3= BUFFER FF
4= MASK P  5= FILTER W
+
----------------------
```

A3.9 Replace Images [S7, L70]

In order to perform more complex processing in sequence, replacement of an image with another image is required. For example, image enhancement applied to G results in F, so that replacing G with the result F allows another successive application.

```
----------------------------------------
*** REPLACE IMAGE-
 1= G WITH F (PACKED, M,N -> MM,NN)
 2= G WITH FF
 3= FF WITH G
 4= FF WITH F (IMBED)
 5= FF WITH P
 6= EXIT TO MENUE
 +
----------------------------------------

------------- F => G ------------------------
      CALL SUBPACK(F,MM,NN,G,MN,MM,NN)
------------- FF => G -----------------------
      CALL MATMOV(G,M,N,FF)
------------- G => FF -----------------------
      CALL MATMOV(FF,M,N,G)
------------- F => FF -----------------------
      CALL IMBED(F,MM,NN,1,MM,1,NN,FF,M,N,L+1,L+1)
------------- P => FF -----------------------
      CALL MATMOV(FF,M,N,P)
```

A3.10 Add Noise

This is the option to add random noise to the original image G or buffer (blurred) image FF. The noise is Gaussian, characterized with mean 0 and standard variance SIGMA. An odd large integer INT is required as the initial number to generate the random number sequence.

```
--------------------------------
*** ADD NOISE ***
1= TO ORIGINAL IMAGE G
2= TO BUFFER (BLURRED) IMAGE FF
+
------------< SIGMA >-----------
INPUT SIGMA
+
------------< ITN >-------------
INPUT ANY LARGE ODD INTEGER
+
--------------------------------
```

A3.11 Thresholding

The option of thresholding gray-level images into binary ones is implimented. One can select three applications to G, F, and FF, and two modes of threshold selection, automatic and manual. The automatic threshold selection is base d on the least-squares and discriminant criterion (A1).

```
-----------------------------------------
 **** THRESHOLDING *****
 1= G TO FF
 2= F TO F
 3= FF TO FF
 4= EXIT TO MENU
 +
-----------------------------------------
 *** THRESHOLDING FOR ID= I8 ***
 GRAY LEVELS = [1, I4]
 0( ) FOR [1, LTH], 1(*) FOR [LTH+1, I4]
 1=AUTO, 2=MANU
 +
--------< For 2= Manual >----------------
 INPUT LTH
 +
-----------------------------------------
```

A3.12 Edge Detection

This option includes a new family of nonlinear edge detectors (A2) based on the blurring-invariant histogram moments and some other usual ones. All of those are characterized by directional homogenuity property. The size of the running local window, (2L + 1) by (2L + 1), is parametrized

by L, with an exception that the Sobel operator is defined only for 3 by 3 window.

```
-----------------------------------------
 *** EDGE DETECTION ***
 INPUT 1=<L<5 FOR MASK SIZE (2L+1)**2
 (QUIT FOR L=0)
 +
-----------------------------------------

      CALL EDGEB(G,M,N,L,F,MM,NN,K)
-------------------------------------------------------------
c   U and X: Blurring-Invariant Histogram-Moments,
c   |x|: absolute value of x, and [x]=x for x>0, =0 for x<=0.
-------------------------------------------------------------
 1 = U2
 2 = U3
 3 = X4
 4 = U2**1/2
 5 = [+U3]**1/3
 6 = [-U3]**1/3
 7 = |U3|**1/3
 8 = |X4|**1/4
 9 = [U2**1/2 - W*|U3|**1/3]
10 = [U2**1/2 - W*|X4|**1/4]
11 = [U2**2 - W*X4]**1/4

12= LAPLACIAN = |g-A|
13= TONYS LAPLACIAN = [g-A]
14= SOBEL (3X3)

+
-----------------------------------------
1=ANOTHER EDGE FILTER
2=MENUE
+
-----------------------------------------
```

A3.13 Filterings

Some types of nonlinear filtering with (2L + 1) by (2L + 1) running local window are implimented as experimental versions for image enhancement.

```
--------< Filtering: G into F >-------------
 1=ENHANCEMENT
```

```
 2=P-MEANS
 3=DIRECTIONAL SMOOTHING
-----------------------------------------
 MASK SIZE L?
 +
-----------------------------------------

---------< For 1= ENHANCEMENT>----------------------
      CALL ENHFLT(G,M,N,L,F,MM,NN)
-------------------------------------------------------
C   For 2 through 4, minimum values of divisors,
C   local average AV and standard deviation SD,
C   are inquired.
-------------------------------------------------------
 1: f=g-AV
 2: f=g/AV¬
 3: f=[(g-AV)/SD]
 4: f=[(g-AV)/SD] + AV
 +
-------------------------------------------------------

---------< For 2= P-MEANS >-------------------------
      CALL PMFLT(G,M,N,L,P,F,MM,NN)
-------------------------------------------------------
C   P-MEAN FILTER   (N.Otsu, 24-Aug.¬83)
C     f=[(g1**P+....+gn**P)/n]**(1/P)
CI  P= parameter for averaging (|P|<=10)
c       f= max(gi) as P tends to + infinit
c       f= usual arithmetic mean for P=1
c       f= 1 as P tends to 0
c       f= harmonic mean for P=-1
c       f= min(gi) as P tends to - infinit
-------------------------------------------------------
 INPUT P
 +
-------------------------------------------------------

---------< 3= DIRECTIONAL SMOOTHING>----------------
      CALL DSMOOTH(G,M,N,L,F,MM,NN,EV,2*L+1,R,4*L+1)
-------------------------------------------------------
```

A3.14 MRAF

MRAF is for the automatic design of Filters based on the Multiple Regression Analysis. For detail refer to ref. A3. The basic idea is to

enable the machine (computer) to learn (design) automatically the optimum filter that is implicitly characterized by a pair of input and ideal output images and evaluated in the sense of least mean square error. It necessitates three images: an original input image G/FF, an ideal output image FF/G, and an image, (mask) P for weighting the learning process. The resultant filter (local mask) is automatically designed in W in the learning process based on MRA. The output image approximated by the filter W is obtained in F by filtering (applying W to) G/FF.

In the case of image restoration in degrading transmission channels, for example, G is a original input (test) image, and FF is a degraded (blurred) output image obtained after transmission. Then, the estimation (learning) of W in the direction from G to FF results in the estimation of the degrading process (filtering) itself, and contrarily, the estimation of W in the direction from FF to G acts as an automatic way of estimating the restoration filter.

This session must be initiated with defining the three images, G, FF, and P. It is noted that the three images might be three sub-areas localized in corresponding positions.

```
----------< L5: MRAF >-------------------------------------
 *** MRAF *** SELECT-
 1= IDEAL OUTPUT/BLURRING (FF)
 2= DESIGN OF LEARNING MASK (P)
 3= ESTIMATION (LEARNING) OF FILTER W (G TO FF  FF TO G)
    (G,FF,and P must have already been defined.)
 4= FILTERING (OPERATION OF W) (TO G  TO FF)
 5= PRINT OBTAINED W
 6= EXIT TO MENUE
 +
 ------------------------------------------------------------

 =======< For 1= MANUAL DESIGN of IDEAL OUTPUT/BLURRING >======
 *** IDEAL OUTPUT/BLURRING (G TO FF) ***
 SELECT-
 1= MANUAL FILTER-DESIGN
 2= QUIT
 +
------------------------------------------------------------
   GOTO L5 (for 2= QUIT)
-------------< 1= MANUAL FILTER-DESIGN >-----------------
 INPUT 1=<L<5 FOR FILTER SIZE (2L+1)**2
   +
   INPUT BLURRING FILTER ROW-WISE WITH DELIMITTING BY COMMAS
        CALL READMSK(W,LW,LW)
```

```
 +, +, ..., +
 ...........
 +, +, ..., +
--------------< PRINTOUT OF W >----------------------
      CALL PRIMSK(W,LW,LW,BLURRING FILTER ,16)
--------------< BLURRING FILTERING >-----------------
      CALL LFT(G,M,N,FF,LB,W,LD)
--------------< DISPLAY OF FF >----------------------
      CALL NORPKM(FF,M,N,H,NL,NB)
=============================================================
   GOTO L5

=======< For 2= DESIGN OF LEARNING MASK P >===============
 *** LEARNING MASK P ***
 1= NEW MASK
 2= OLD ONE
 +
===========<For 1= NEW MASK >=====================
---------------- INITIALIZE ---------------------
      CALL MATZER(P,M,N)
-------------------------------------------------
 DESIGN OF LEARNING MASK
 1= NO MASK (= UNIFORM ALL OVER IMAGE)
 2= FROM RESULT IMAGE F>=0
 3= FROM BUFFER IMAGE FF
 4= EXIT
  +
-------< For 1 >---------------------------------
      CALL MATCON(P,M,N,1.0)
-------< For 2 >---------------------------------
      CALL IMBED(F,MM,NN,1,MM,1,NN,P,M,N,L+1,L+1)
-------< For 3 >---------------------------------
      CALL MATMOV(P,M,N,FF)

=================================================
 1= DISPLAY OF MASK P
 2= MODIFICATION OF BINARY P
 3= COPY P TO FF
 4= EXIT
+
-------< For 1=DISPLAY >-------------------------
CALL NORPKM(P,M,N,H,NL,MDSG)
-------< For 2= MODIFICATION >--------------------
CALL TRANS1(P,M,N,EV,R,IEX)
-------< For 3= COPY >---------------------------
```

```
      CALL MATMOV(FF,M,N,P)
=============================================================
   GOTO L5

==========< For 3= MRA-ESTIMATION OF FILTER W >==================
*** MRA-ESTIMATION OF FILTER W ***
SELECT
1= ESTIMATION OF FILTER (G TO FF)
2= ESTIMATION OF FILTER (FF TO G))
+
INPUT 1=<L<5 FOR FILTER SIZE (2L+1)**2
-+
 ----< 2= ESTIMATION OF (REBLURRING) MASK (FF TO G) >---------
-     CALL MRAFDM(FF,M,N,G,P,LE,R,A,W,LD,B2,BA,BO,O2,OA,CRC)
 ----< 1= ESTIMATION OF (BLURRING) MASK (G TO FF) >-----------
=     CALL MRAFDM(G,M,N,FF,P,LE,R,A,W,LD,B2,BA,BO,O2,OA,CRC)
 ================ ============================================
   GOTO L5
=
=====< For 4= FILTERING (APPLICATION OF W TO FFG) >=======
*** FILTERING (APPLICATION OF FILTER W TO GFF)
RESULT IS STORED INTO F ****
SELECT-
1= TO G
2= TO FF
-+
 --------< 1 >-----------------------------
-     CALL LFTP(G,M,N,W,LE,LD,F,MM,NN)
 --------< 2 >-----------------------------
-     CALL LFTP(FF,M,N,W,LE,LD,F,MM,NN)
 --------< DISPLAY >-----------------------
=     CALL NORPKM(F,MM,NN,H,NL,NO)
 =============================================================
   GOTO L5
=
======< For 5= OUTPUT OF OBTAINED MASK W>================
=     CALL PRIMSK(W,LW,LW,OBTAINED MASK,13)
 =============================================================
   GOTO L5
```

A3.15 KL-Filtering

This block is concerned with the (statistically) orthogonal decomposition of a grey-level image. Unlike the usual application of Karhunen-Loeve expansion, the statistics (auto-correlation or covariance matrix) is

computed for local neighbourhoods all over the image. Therefore the resulting eigenvectors provide a set of mutually orthogonal local filters which are ordered in series with respect to "degree of dominance" evaluated by the eigenvalues.

The block consists of three modes: The first one, that should be initiated, is the selection of the size L of local neighbour, (2L + 1) by (2l + 1), and the type of the statistics. The second is the eigen projection, or the filtering by using the eigen filters (eigenvectors). The third is an optional edge detection based on the eigen projections.

```
-------< L3 >------------------------
 *** KL-FILTERING ***
 1= NEW SIZE, STAT,EIGEN
  2= EIGEN PROJECTION
  3= KL-EDGE DETECTION
 4= MENUE
 +
-------------------------------------

======< For 1= CALCULATION OF EIGEN-MASKS >=====================
-------------< SET PARAMETERS L, IST >--------------------
 LOCAL MASK SIZE L (=1,...,4)?
 +
 TYPE OF STATISTICS? (1=AUTO-CORR., 2=COVAR.)
 +
-------------< EIGEN PROBLEM >-----------------------------
      CALL KLEIGEN(G,M,N,L,LD,IST,R,A,EV,W)
---< PRINTOUT OF EIGENVALUES AND EIGEN-MASKS (VECTORS) >---
      CALL EIGENR(IUN,L,IST,EV,W,LD)
================================================================
      GOTO L3

======< For 2=  EIGEN PROJECTION (FILTERING) >==================
-------< SELECTION OF A (WEIGHTED) EIGEN-MASK W >----------
      CALL EIGENW(EV,JTH,A,W,LD)
--------< PROJECTION (FILTERING) >-------------------------
      CALL LFTP(G,M,N,W,L,LD,F,MM,NN)
--------------< OUTPUT >-----------------------------------
      CALL NORPKM(F,MM,NN,H,NL,NID)
      CALL MATPRI(A,LD,1,EIGEN-WEIGHT,12)
      CALL PRIMSK(W,LW,LW,MASK,4)
================================================================
      GOTO L3
```

```
=======< For 3= KL-EDGE >=====================================
 *** ENTER 1 <= JS <= JE <=LD
 +, +
-------------------------------------------------------------
      CALL KLEDGE(G,M,N,F,MM,NN,EV,L,LD,JS,JE)
      CALL NORPKM(F,MM,NN,H,NL,NID)
=============================================================
      GOTO L3
```

A3.16 SVD [S14]

This option produces the Singular Value Decomposition of input gray-level image G. The resultant image F is its truncation up to the Kth order pre-assigned. SVD can be performed to the partitioned (MP by NP) images of G (M by N). The selection such that MP=M and NP=N, therefore, means SVD for the whole image G with no partition.

```
-------------------------------------------------
  PARTITION MATRIX SIZE MP, NP?
 +, +
-------------------------------------------------
  UP TO WHICH ORDER K < MIN(MP,NP)?
 +
-------------------------------------------------
      CALL IMBED(G,M,N,I1,MP,J1,NP,FF,MP,NP,1,1 )
      CALL ANORM(FF,MP*NP,V)
      CALL SVDNO2(FF,MP,NP,EV,R,W,K,1.0E-7)
      CALL MATMLT(3,EV,MP,K,R,NP,K,FF,MP,NP)
      CALL IMBED(FF,MP,NP,1,MP,1,NP,F,MM,NN,I1,J1)
-------------------------------------------------
```

A3.17 END [S15]

This selection stops the program and ends the session.

A4. DOCUMENTATION OF SUBROUTINES

A4.1 NORPKM

```
      SUBROUTINE NORPKM(G,M,N,H,L,NID)
=============================================================
C   OPTIONAL DISPLAY OF IMAGE TO NORPAK
C   USING SUB.NORPK N.OTSU, FEB.16,1983
C   * Intended not to alter G and NID
```

```
------------------------------------------------------------------
CI  G = given real image in any scale
CI  M= row-size (number of columns)
CI  N= number of rows
CO  H = histogram in L levels
CI  L= number of gray-levels for output, [EPS,L-EPS] (L <= 300)
C       all output image are stretched to this ful-range
CI  NID = image identification number
==================================================================
      DIMENSION G(M,N),H(L),HB(300)

------------------------------------------
 SELECT -
 1= output as is (in ful-range)
 2= gray-scale inversion
 3= clipping negatives
 4= clipping positives & inversion
 5= clipping at upper and lower percentiles
 6= thresholding
 7= quit
 +
---------< ISSUE OF ID and SELECTION >-------
 ID = I8,    OUT = I2
------------------------------------------
=====< MIN-MAX DETECTION for 1,2,3,4 >=======
    VMIN, VMAX = ,2F10.3

------------------------------------------
    1 CALL NORPK(G,M,N,VMIN,VMAX,1)
      RETURN
------------------------------------------
    2 CALL NORPK(G,M,N,VMIN,VMAX,-1)
      RETURN
------------------------------------------
    3 CALL NORPK(G,M,N,0.0,VMAX,1)
      RETURN
------------------------------------------
    4 CALL NORPK(G,M,N,VMIN,0.0,-1)
      RETURN

======< for 5,6,7 >=======================
      CALL NRANGE(G,M,N,L)
      CALL NHIST(G,M,N,L,H,HMAX)
```

```
----------< PERCENTILE CUT >----------------
 LOWER CUTOFF PERCENTILE (% >=0 )?
 +
-----------------------------------------------
 UPPER CUTOFF PERCENTILE (% >=0 )?
 +
-----------------------------------------------
      CALL NORPK(G,M,N,GPL,GPU,1)
      RETURN

--------< THRESHOLDING >----------------------------
      CALL HISTOP(H,L,450,350,LST,25,HMAX10.0,10,0)
-----------------------------------------------------
 *** THRESHOLDING FOR ID=,I8, ***
 GRAY LEVELS = [1,,I4,]
 0( ) FOR [1, LTH], 1(*) FOR [LTH+1,,I4,]
 1=AUTO, 2=MANU
 +
-----------< for 2 >-------------------------
 INPUT LTH
 +
-----------< for 1 >-----------------------------------
      CALL THRESA(H,L,AVE,VAR,LTH,P0,P1,U0,U1,SEP,HB)
-------------------------------------------------------
*** THRESA ***

     TOTAL AVE. OF LEVEL = ,F7.1
     TOTAL VAR. OF LEVEL = ,F7.1
     OPTIMAL THRESHOLD LEVEL OBTAINED LTH = ,I4
     CLASS PROB. P0 = ,F5.1
                 P1 = ,F5.1
     " AVE. LEV. U0 = ,F5.1
                 U1 = ,F5.1
     SEP. MEASURE   = ,F5.3
-------------------------------------------------------
      CALL HISTOP(HB,L,450,100,LST,15,0.1,10,0)
-------------------------------------------------------
      CALL NORPK(G,M,N,TH,TH,1)
      RETURN
```

A4.2 NORPK

```
          SUBROUTINE NORPK(F,MD,ND,A,B,IP)
     ================================================================
     C      SUBROUTINE TO OUTPUT IMAGE TO NORPAK COLOUR DISPLAY
     C      N.OTSU JUNE-8,83
```

```
C-------------------------------------------------------------
CI      F(MD,ND)= OUTPUT IMAGE
CI      MD= NUMBER OF PIXELS IN ROW
CI      ND= NUMBER OF ROWS
CI      A = LOWER LIMIT OF RANGE
CI      B = UPPER LIMIT OF RANGE
CI      IP= SWITCH 1=POSI, -1=NEGA, 0=ERASE
C-------------------------------------------------------------
CW      I1= STARTING COLUMN ON NORPAK MAP IMAGE
CW      J1= STARTING ROW ON NORPAK MAP IMAGE
CW      KS= SCALE FACTOR FOR ENLARGEMENT OF F
CW      NB= THE NUMBER OF BANK TO LOAD THE IMAGE F (NB=1,2,3)
C=============================================================
      DIMENSION F(MD,ND)
      INTEGER*2 IBUFF(640)

C---------------< PARAMETERS SETTING >---------------------
 *** OUTPUT TO NORPK ***
 STORRING BANK [1,2,3]?, 4= QUIT (EXIT)
 +
C-----------------------------------------
 STARTING COLUMN ON NORPAK [1-640]?
 +
C-----------------------------------------
 STARTING ROW ON NORPAK [1-480]?
 +
C-----------------------------------------
 SCALING FACTOR [1,2,...]?
 +
C-----------------------------------------
 ARE YOU SURE ?  1=YES, 2=NO
 +
C-----------------------------------------

C---------< WRITE TO NORPAK >---------------
      RETURN
```

A4.3 The Other Subroutines

```
      SUBROUTINE EDGEB(G,M,N,L,F,MM,NN,K)
C=============================================================
C   EDGE DETECTORS / N.OTSU (FEB.16, REVISED JUNE 10, 1983)
C-------------------------------------------------------------
C  G(M,N)= input image
C  M= row-size of G
```

```
C  N= column-size (number of rows) of G
C  F(MM,NN)= output image (PACKED)
C  MM= row-size of F
C  NN= column-size (number of rows) of F
C  L= operator-mask size (2L+1)X(2L+1)
C  K=type of operator (K=1,...,14)
c    1 = U2
c    2 = U3
c    3 = X4
c    4 = U2**1/2
c    5 = [+U3]**1/3
c    6 = [-U3]**1/3
c    7 = |U3|**1/3
c    8 = |X4|**1/4
c    9 = [U2**1/2 - W*|U3|**1/3]
c   10 = [U2**1/2 - W*|X4|**1/4]
c   11 = [U2**2 - W*X4]**1/4
c   12= LAPLACIAN = |g-A|
c   13= TONYS LAPLACIAN = [g-A]
c   14= SOBEL (3X3)
c   where U and X: Blurring-Invariant Histogram-Moments,
c   |x|: absolute value of x, and [x]=x for x>0, =0 for x<=0.
C=================================================================

      SUBROUTINE IMREAD(IUN,ID,JD,L,IS,II,JS,JJ,F,KSW)
    C=================================================
    C   READ ANY PORTION OF IMAGE FROM DATA FILE
    C   (N.OTSU  JAN.13, 1983)
    C-------------------------------------------------
    CI  IUN = logical unit number for input
    CI   ID = number of columns of data image
    CI   JD = number of rows of data image
    CI    L = number of gray levels
    CI   IS = the starting column to read
    CI   JS = the starting row to read
    CI   II = number of columns for cutout image
    CI   JJ = number of rows for cutout image
    CO   F(II,JJ) = real matrix for cutout image
    CI   KSW = 1 for direct, = 2 for negating read
    C=================================================

          SUBROUTINE KLEDGE(G,M,N,F,MM,NN,A,L,LL,JS,JE)
    C=================================================
    C  EDGE DETECTION BASED ON KL-FILTER (N.OTSU)
```

```
C  f= SQRT[squared sum of KL-eigen images from JS to JE
C-------------------------------------------------------
CI  G(M,N)= input image
CI  M= row-size of G
CI  N= number of rows og G
CO  F(MM,NN)= output image of edge
CI  MM= row-size of F
CI  NN= number of rows of F
CI  A(LL,LL)= KL-eigenvector matrix for G
CI  L= size of local mask (2*L+1)**2=LL
CI  LL= (2*L+1)**2 = dimenison of the mask and A
CI  JS= the starting order of eigenvector for adoption
CI  JE= che ending order of eigenvector for adoption
C=======================================================


      SUBROUTINE LFT(G,M,N,F,L,W,LD)
C=============================================================
C   LINEAR FILTERING OF IMAGE G WITH A LOCAL MASK W  (UNPUCKED)
C-------------------------------------------------------------
CI  G(M,N)= input gray-level image
CI  M= row-size of matrix G
CI  N= column-size (number of rows) of matrix G
CO  F(M,N)= resultant image (leaving the margins of width L)
CI  W(LD) = local mask (row-wise)
CI  L= size of the local mask (2*L+1)**2  (L=1,2,...)
CI  LD= dimension of W ( LD=(2*L+1)**2 )
C=============================================================


      SUBROUTINE LFTP(G,M,N,W,L,LD,F,MM,NN)
C=============================================================
C   LINEAR FILTERING OF IMAGE G WITH A LOCAL MASK W (PACKED)
C-------------------------------------------------------------
CI  G(M,N)= input gray-level image
CI  M= row-size of matrix G
CI  N= column-size of matrix G
CI  W(LD) = local mask given
CI  L= size of the local mask (2*L+1)**2  (L=1,2,...)
CI  LD= dimension of W (LD=(2*L+1)**2)
CO  F(MM,NN)= resultant image
CI  MM= row size of F (MM=M-2*L)
CI  NN= column size of F (NN=N-2*L)
C=============================================================
```

```
      SUBROUTINE LSTA(F,M,N,L,IS,R,A,W,LL)
C=================================================================
C  COMPUTE LOCAL NEGHBOURHOOD STATISTICS OF AN IMAGE
C  N.Otsu, Feb.-83
C-----------------------------------------------------------------
CI  F(M,N)= input image
CI  M= row-size
CI  N= column-size (number of rows)
CI  L= size of local window (2*L+1)**2  (L=1,2,...)
CI  IS= switch for second order statistical matix R
c       1= auto-correlation, 2= covariance
CO  R(LL,LL)= neghbourhood auto-correlation/covariance matrix
CO  A(LL)= neghbourhood average (in row-wise ordering)
CW  W(LL)= local area of F cut out by the window (in row-wise ordering)
CI  LL=  dimension for R, A, and W;  LL=(2*L+1)**2
C=================================================================


            SUBROUTINE MATCON(A,M,N,C)
          C=====================================================
          C   CREATE CONSTANT-VALUED MATRIX  /N.OTSU, 23-SEP-83
          C-----------------------------------------------------
          CO  A(M,N)= matrix to be filled with a constant value C
          CI  M= row-size of A
          CI  N= number of rows in A
          CI  C= a constant value to fill A
          C-----------------------------------------------------


      SUBROUTINE MRAFDM(G,M,N,F,P,L,R,C,W,LL,GG,GA,FG,FF,FA,CRC)
C=================================================================
C    FILTER DESIGN BY MULTIPLE REGRESSION ANALYSIS /N.OTSU, 23-SEPT-83
C    The learning can be controled from outside by forced weighting mask.
C-----------------------------------------------------------------
CI  G(M,N)= (blurred) image to be filtered
CI  M=  row-size (number of columns) of G and F
CI  N=  column-size (number of rows) of G and F
CI  F(M,N)= forced reference image (desired ideal output)
CI  P(M,N)= forced weighting image (mask) for learning, P>=0
CI  L=  local-mask size (L=1,2,3,..)
CW  R(LL,LL)= local auto-correlation matrix of G, finally A**2*R**-1
CW  C(LL)=  partial-correlation vector between local G and f, finally C/A
CO  W(LL)=  optimal local-mask to be obtained
CI  LL=  size of R, C, and W ( LL >= (2*L+1)**2 )
CO  GG=  power of G (averaged g*g)
```

```
CO  GA=  average of g
CO  FG=  co-power between F and G (averaged g*f)
CO  FF=  power of F (averaged f*f)
CO  FA=  average of f
CO  CRC=  Ct*invR*C
C=================================================================

      SUBROUTINE NHIST(G,M,N,L,H,HMAX)
C=================================================================
C  COMPUTE NORMALIZED HISTOGRAM OF AN IMAGE (N. OTSU, FEB.4,¬83)
C-----------------------------------------------------------------
CI  G(M,N)= input image
CI  M= row-size
CI  N= column-size (number of rows)
CI  L= number of gray-levels
c       gray-levels are supposed to be within [0, L-1].
c       Otherwise, subr. NRANGE must be called beforehand.
CO  H(L)= array for histogram of G; normalized as
c         H(I) >= 0,    H(1) + ... + H(L) = 1c
c         H(I) for the gray-levels K-1 <= g < K
CO  HMAX= maximum value of H(I) for I.
C=================================================================

      SUBROUTINE NRANGE(G,M,N,L)
C=================================================
C   NORMARIZATION OF GRAY SCALE RANGE  (N.OTSU, 2,FEB.´83)
C   The gray-scale range of given image G(M,N) is linearly
C   rescaled into [EPS, L-EPS], so that the trancated
C   integral values [G(I,J)] fall on [0, L-1].
C-------------------------------------------------
CI  G(M,N)= input image
CI  M= row-size of G
CI  N= number of rows of G
CI  L= number of gray-levels wanted
C=================================================

      SUBROUTINE NSTD(A,M,N,AVE,SD)
C=========================================
C   NORMILIZE THE DATA GIVEN IN A MATRIX
C   FORM A(M,N) INTO THE STANDARD VARIABLES,
C   I.E. MEAN=0 AND STANDARD DEVIATION=1
C-----------------------------------------
```

```
CI  A(M,N)= data matrix
CI  M= row-size of A
CI  N= column-size of A
CO  AVE= average value of A
CO  SD= standard deviation of A
C=========================================


      SUBROUTINE PMFLT(G,M,N,L,P,F,MM,NN)
C=============================================
C   P-MEAN FILTER   (N.Otsu, 24-Aug.-83)
C     f=[(g1**P+....+gn**P)/n]**(1/P)
C=============================================
CI  G(M,N)= input gray-level image (G>=0)
CI  M= row-size of matrix G
CI  N= column-size of matrix G
CI  L= size of local mask n=(2*L+1)**2  (L=1,2...)
CI  P= parameter for averaging (|P|<=10)
c       f= max(gi) as P tends to + infinit
c       f= usual arithmetic mean for P=1
c       f= 1 as P tends to 0
c       f= harmonic mean for P=-1
c       f= min(gi) as P tends to - infinit
CO  F(MM,NN)= resultant image
CI  MM= row-size of F (MM=M-2*L)
CI  NN= column-size of F (NN=N-2*L)
C=============================================


     SUBROUTINE PRIMSK(A,M,N,-HEAD-,MOJI)
C=================================================================
C   PRINTOUT OF A MASK/MATRIX WHICH IS STORED ROW-WISE
C   (N.OTSU 24-MAR-83)
C-----------------------------------------------------------------
CI  A(M,N)= mask/matrix to printout
CI  M= number of columns (row-size)
CI  N= number of rows (column-size)
CI  HEAD= header for title
CI  MOJI= number of characters in HEAD
C=================================================================


          SUBROUTINE READMSK( A, M, N )
      C=================================================
      C   INPUT VALUES OF LOCAL MASK ROW-WISE FROM TERMINAL
```

```
C   FOR THE PURPOSE OF MANUAL DESIGN OF LINEAR FILTER
C   (N.OTSU 24-MAR-83)
C-------------------------------------------------------
CO  A(M,N)= local mask to be designed
CI  M= row-size of A
CI  N= number of rows of A
C=======================================================
```

```
      SUBROUTINE THRESH(H,L,A,V,LT,P0,P1,A0,A1,SEP)
C=============================================================
C  AUTOMATIC THRESHOLD SELECTION FROM GRAY-LEVEL HISTOGRAMS
C  By N. Otsu (Cf. IEEE Trans. SMC-9, 1, 62-66, Jan. 1979, also
C  Proc. 4th IJCPR, 592-596, Nov. 1979)
C-------------------------------------------------------------
CI  H(L)= normalized gray-level histogram;  H(1)+H(2)+...+H(L)=1
CI  L= number of gray levels;  [1,2,...,L]
CO  A= total mean of gray levels
CO  V= total variance of gray levels
CO  LT= optimal threshold level; C0=[1,...,LT] and C1=[LT+1,...,L]
CO  P0= Prob.(C0)
CO  P1= Prob.(C1);  P0 + P1 = 1
CO  A0= mean level of Class C0
CO  A1= mean level of Class C1
CO  SEP= separability measure;  0 <= SEP <= 1
C=============================================================
```

REFERENCES

A1. N. Otsu. A threshold selection method from gray-level histograms." IEEE Trans. Systems, Man Cybern. SMC-9(1): 62-66; 1979.

A2. N. Otsu and T. Kasvand. A new family of nonlinear edge detectors for noisy images. Proc. 27th SPIE Int. Conf., San Diego, CA. Aug. 1983. pp. 70-75.

A3. N. Otsu. Multiple regression analysis approach to the automatic design of adaptive image processing systems. Proc. 27th SPIE Int. Conf., San Diego, CA, Aug. 1983. pp. 10-17.

APPENDIX B. SOURCE LIST OF GIPSN.FOR

```
C **** GIPSN (General Image Processing System on NORPAK) *****
C **** N.OTSU (Feb.-7, / April-6, reviced June-11 1983) ********
      PARAMETER IJD=400,MN=IJD*IJD,NL=256,LL=100,LL2=LL*LL
      DIMENSION F(MN),G(MN),EV(LL2),W(LL),R(LL2),A(LL)
      DIMENSION FF(MN),P(MN),H(NL),HB(NL)
      INTEGER*2 ILTP(NL),ILTI(NL),ILTD(NL),ILTT(NL)
C-----------------------------------------------------------------
C  G(M,N)= input image data (M columns and N rows)
C  F(MM,NN)= output image processed
C  FF(M,N)= image buffer
C  P(M,N)= image buffer mainly for forced weighting mask
C  M, N = number of columns, and number of rows of images G and FF
C  MM, NN = number of columns, and number of rows of images F
C  NL= number of gray-levels of images
C  IUN= unit number for reading image data
C  IPN= switch for positive or negative read of image
C  NID= identification number of images
C  L= local mask size ( (2*L+1)**2 <= LL )
C  R(LL*LL)= local statistics
C  EV(LL*LL)= eigenvectors for local statistics
C  W(LL), A(LL)= working areas, e.g. eigenvalues
C  H(NL)= normalized histogram
C  HB(NL)= criterion function for threshold selection
C  ILTP, ILTI, TLTD, ILTT(NL)= look-up tables for colour display
C  ITERM= 1=TEKTRONIX (graphic terminal), 0=others
C*****************************************************************

      ITERM=0
      PRINT*,'1= TEKTRONIX terminal'
      PRINT*,'2= other terminals'
      READ(5,*) ISW
      IF(ISW.EQ.1) ITERM=1
      CALL INITT(480)
      GOTO 20

C*** SELECTION ************************************************************
  111 CALL CLEAR
      WRITE(6, 112)
  112 FORMAT(1H ,'========== MENUE ================='/
     &'  1= READ IMAGE DATA (TO G, L1)'/
     &'  2= IMAGE DATA NORMALIZATION (TO G/FF,L21)'//

     &'  3= NORPAK INITIALIZATION (L20)'/
```

```
     &'  4= COLOUR CONTROL (L25)'/
     &'  5= DISPLAY ERASING (L22)'//

     &'  6= DISPLAY IMAGES (G,F,FF,P, L90)'/
     &'  7= REPLACE IMAGES (G/FF WITH F/FF, L70)'/
     &'  8= ADD NOISE (TO G/FF, L6)'/
     &'  9= THRESHOLDING (G TO FF / F / FF , L97)'//

     &' 10= EDGE DETECTION (G TO F, L4)'/
     &' 11= FILTERINGS (G TO F, L80)'/
     &' 12= MRAF (G&FF&P TO F, L5)'/
     &' 13= KL-FILTERING (G TO F, L3)'/
     &' 14= SVD (G TO F, L85)'/

     &  15= END (L9)'/' ====================================')
      PRINT*,'SELECT MENUE'
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.15) GOTO 111
C           1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15
      GOTO (1,21,20,25,22,90,70, 6,97, 4,80, 5, 3,85, 9),ISW

C**** IMAGE INPUT ************************************************************

C---------------- IUN, IDIM, JDIM, IPN -------------
    1 CALL CLEAR
      IDIM=128
      JDIM=128
      IPN=1
      WRITE(6,100)
  100 FORMAT(1H ,'*** SELECT IMAGE DATA ***'///
     &' 10= MANUAL INPUT '//
     &' 11= GIRL (128X128) '/
     &' 12= COUPLE (128X128) '/
     &' 13= MOON (128X128) '/
     &' 14= AIRPLANE (128X128) '/
     &' 21= GIRL (256X256) '/
     &' 31= CHROMS (640X512) '/
     &' 32= FINGER (400X400) '/
     &' 41= GENERAL (IDXJD) '/
     &' 1= MENUE '//)
      READ(5,*) IUN
      IF(IUN.LT.10) GOTO 111
      IF(IUN.GT.49) GOTO 1
      GOTO(14,12,13,16),IUN/10
   12 IDIM=256
```

```
      JDIM=256
      GOTO 14
   13 IDIM=640
      JDIM=512
      IF(IUN.EQ.31) GOTO 14
      IDIM=400
      JDIM=400
      GOTO 14
   16 PRINT*,'IDIM (ROW-SIZE) ?'
      READ(5,*) IDIM
      PRINT*,'JDIM (NUMBER OF ROWS) ?'
      READ(5,*) JDIM
      PRINT*,'1=POSI-, 2=NEGA- READ ?'
      READ(5,*) IPN

C---------- M, N ----------------------------------
   14 PRINT*,'NUMBER OF COLUMNS YOU WANT ? '
      PRINT*,'<= MIN.',IDIM,IJD
      READ(5,*) M
      PRINT*,'NUMBER OF ROWS YOU WANT ? '
      PRINT*,'<= MIN.',JDIM,IJD
      READ(5,*) N
      IF(IUN.NE.10) GOTO 11

C---------- TERMINAL INPUT ------------------------
      CALL CLEAR
      CALL READBI(G,M,N)
      IDIM=M
      JDIM=N
      IS=1
      JS=1
      GOTO 10

C---------- IS, JS --------------------------------
   11 PRINT*,'STARTING COLUMN ? LE',IDIM-M+1
      READ(5,*) IS
      PRINT*,'STARTING ROW ? LE',JDIM-N+1
      READ(5,*) JS

C---------- READ FROM FILE ------------------------
      CALL IMREAD(IUN,IDIM,JDIM,NL,IS,M,JS,N,G,IPN)

C---------- RAW-HISTOGRAM -------------------------
   15 CONTINUE
      PRINT*,'RAW-HISTOGRAM PLOT: 1=YES, 2=NO ?'
```

```
      READ(5,*) ISW
      IF(ISW.NE.1) GOTO 10
      CALL NHIST(G,M,N,NL,H,HMAX)
      CALL HISTOP(H,NL,450,350,550/NL,25,HMAX/10.0,10,0)
      IF(ITERM.EQ.1) CALL TINPUT(ICH)

C-------------- DISPLAY OF G ----------------------------
      WRITE(6,113) IDIM,JDIM,M,N,IS,JS,IPN
  113 FORMAT(1H ,' IDIM, JDIM,  M, N,  IS, JS,  IPN =',/
     & 7I4)
   10 NID=(IUN*1000+IS)*1000+JS
      CALL NORPKM(G,M,N,H,NL,NID)
      GOTO 111


C**** KL-FILTER *****************************************************
    3 CALL CLEAR
      WRITE(6,300)
  300 FORMAT(1H ,'*** KL-FILTERING ***'//
     &' 1= NEW SIZE, STAT, & EIGEN'/
     &'  2= EIGEN PROJECTION'/
     &'  3= KL-EDGE DETECTION'/
     &' 4= MENUE'//)
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.4) GOTO 3
      GOTO (32,35,301,111),ISW

C===== CALCULATION OF EIGEN-MASKS ================================
C-------- SET PARAMETERS L, IST, ----------------------------
   32 PRINT*,'LOCAL MASK SIZE L (=1,...,4)?'
      PRINT*,' '
      READ(5,*) L
      IF(L.LT.1.OR.L.GT.4) GOTO 32
      LW=2*L+1
      LD=LW*LW
      MM=M-2*L
      NN=N-2*L
   33 PRINT*, 'TYPE OF STATISTICS? (1=AUTO-CORR., 2=COVAR.)'
      PRINT*,' '
      READ(5,*) IST
      IF(IST.LT.1.OR.IST.GT.2) GOTO 33

C------------ EIGEN PROBLEM ----------------------------
      CALL KLEIGEN(G,M,N,L,LD,IST,R,A,EV,W)
```

```
C------ PRINTOUT OF EIGENVALUES AND EIGEN-MASKS (VECTORS) -----
      CALL CLEAR
      CALL EIGENR(IUN,L,IST,EV,W,LD)
      GOTO 3

C===== EIGEN PROJECTION (FILTERING) ==========================
C----- SELECTION OF A (WEIGHTED) EIGEN-MASK W -------------
   35 CALL EIGENW(EV,JTH,A,W,LD)

C----- PROJECTION (FILTERING) -----------------------
      CALL LFTP(G,M,N,W,L,LD,F,MM,NN)

C------------ OUTPUT ---------------------------------
      NID=IUN*100000+(30+L)*1000+IS*100+JTH
      CALL NORPKM(F,MM,NN,H,NL,NID)
      CALL MATPRI(A,LD,1,'EIGEN-WEIGHT',12)
      CALL PRIMSK(W,LW,LW,'MASK',4)
      IF(ITERM.EQ.1) CALL TINPUT(ICH)
      GOTO 3

C====== KL-EDGE ==============================================
  301 WRITE(6,311) LD
  311 FORMAT(1HO,'*** ENTER 1 <= JS <= JE <=',I3/)
      READ(5,*) JS,JE
      CALL KLEDGE(G,M,N,F,MM,NN,EV,L,LD,JS,JE)
      NID=IUN*100000+(300+JS)*100+JE
      CALL NORPKM(F,MM,NN,H,NL,NID)
      GOTO 3


C *** EDGE DETECTION ***************************************************
    4 CALL CLEAR
      PRINT*,'*** EDGE DETECTION ***'
      PRINT*,'INPUT 1=<L<5 FOR MASK SIZE (2L+1)**2'
      PRINT*,'(QUIT FOR L=0)'
      READ(5,*) L
      IF(L.EQ.0) GOTO 111
      IF(L.LT.1.OR.L.GT.4) GOTO 4
      MM=M-2*L
      NN=N-2*L
C---------------------------------------
   45 CALL EDGEB(G,M,N,L,F,MM,NN,K)
C---------------------------------------
      NID=IUN*10000+(40+L)*100+K
      CALL NORPKM(F,MM,NN,H,NL,NID)
```

```
C-----------------------------------------
      PRINT*,'1=ANOTHER EDGE FILTER'
      PRINT*,'2=MENUE'
      PRINT*,' '
      READ(5,*) ISW
      IF(ISW.EQ.1) GOTO 45
      GOTO 111


C***** MRAF ************************************************************
    5 CALL CLEAR
      WRITE(6,500)
  500 FORMAT(1H ,'*** MRAF *** SELECT-'/
     &' 1= IDEAL OUTPUT/BLURRING (FF)'/
     &' 2= DESIGN OF LEARNING MASK (P)'/
     &' 3= ESTIMATION (LEARNING) OF FILTER W (G TO FF / FF TO G)'/
     &'    (G,FF,and P must have already been defined.)'/
     &' 4= FILTERING (OPERATION OF W) (TO G / TO FF)'/
     &' 5= PRINT OBTAINED W'/
     &' 6= EXIT TO MENUE'//)
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.6) GOTO 5
      GOTO (52,53,544,56,55,111),ISW

C======= IDEAL OUTPUT/BLURRING (G TO FF) ======================
   52 WRITE(6,520)
  520 FORMAT(1H ,'*** IDEAL OUTPUT/BLURRING (G TO FF) ***'/
     &' SELECT-'/
     &' 1= MANUAL FILTER-DESIGN'/
     &' 2= QUIT'/)
      READ(5,*) ISW
      IF(ISW.NE.1) GOTO 5

C--------------- MANUAL DESIGN ----------------------------
  521 PRINT*,'INPUT 1=<L<5 FOR FILTER SIZE (2L+1)**2'
      PRINT*,' '
      READ(5,*) LB
      LW=2*LB+1
      LD=LW*LW
      MMM=M-2*LB
      NNN=N-2*LB
      PRINT*,'INPUT BLURRING FILTER ROW-WISE WITH DELIMITTING BY COMMAS'
      CALL READMSK(W,LW,LW)
      PRINT*,' '
      CALL PRIMSK(W,LW,LW,'BLURRING FILTER ',16)
```

```
C---------------- FILTERING ------------------------
      CALL LFT(G,M,N,FF,LB,W,LD)

C---------------- DISPLAY OF FF --------------------
   51 NB=IUN*1000+510+LB
      CALL NORPKM(FF,M,N,H,NL,NB)
      GOTO 5

C======= DESIGN OF LEARNING MASK P ====================
   53 CONTINUE
      PRINT*,'*** LEARNING MASK P ***'
      PRINT*,'1= NEW MASK'
      PRINT*,'2= OLD ONE'
      READ(5,*) ISW
      IF(ISW.EQ.2) GOTO 545

C------------ INITIALIZE ----------------------------
      CALL MATZER(P,M,N)
C-------------------------------------------------------
  540 PRINT*,'DESIGN OF LEARNING MASK'
      PRINT*,'1= NO MASK (= UNIFORM ALL OVER IMAGE)'
      PRINT*,'2= FROM RESULT IMAGE F>=0'
      PRINT*,'3= FROM BUFFER IMAGE FF'
      PRINT*,'4= EXIT'
      READ(5,*) MDSG
      IF(MDSG.LT.1.OR.MDSG.GT.4) GOTO 540
      GOTO (541,542,543,5),MDSG
C--------------------------------------
  541 CALL MATCON(P,M,N,1.0)
      GOTO 545
C-----------------------------------------------
  542 CALL IMBED(F,MM,NN,1,MM,1,NN,P,M,N,L+1,L+1)
      GOTO 545
C-----------------------------------------------
  543 CALL MATMOV(P,M,N,FF)
      GOTO 545
C-----------------------------------------------
  546 CALL TRANS1(P,M,N,EV,R,IEX)
C-----------------------------------------------
  545 PRINT*,'1= DISPLAY OF MASK P'
      PRINT*,'2= MODIFICATION OF BINARY P'
      PRINT*,'3= COPY P TO FF'
      PRINT*,'4= EXIT'
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.4) GOTO 545
```

```
      GOTO ( 591,546,580,5), ISW
C-------------------------------------------------
  580 CALL MATMOV(FF,M,N,P)
      GOTO 5
C-------------------------------------------------
  591 CALL NORPKM(P,M,N,H,NL,MDSG)
      GOTO 545

C========== MRA-ESTIMATION OF FILTER W ====================
  544 CONTINUE
      WRITE(6,530)
  530 FORMAT(1H ,'*** MRA-ESTIMATION OF FILTER W ***'/
     &' SELECT'/
     &' 1= ESTIMATION OF FILTER (G TO FF)'/
     &' 2= ESTIMATION OF FILTER (FF TO G)'//)
      READ(5,*) ISWR
      PRINT*,'INPUT 1=<L<5 FOR FILTER SIZE (2L+1)**2'
      PRINT*,' '
      READ(5,*) LE
      LW=2*LE+1
      LD=LW*LW
      IF(ISWR.EQ.1) GOTO 54
C------ ESTIMATION OF (REBLURRING) MASK (FF TO G) ---------
      CALL MRAFDM(FF,M,N,G,P,LE,R,A,W,LD,B2,BA,BO,O2,OA,CRC)
      GOTO 55
C------ ESTIMATION OF (BLURRING) MASK (G TO FF) -----------
   54 CALL MRAFDM(G,M,N,FF,P,LE,R,A,W,LD,B2,BA,BO,O2,OA,CRC)

C======= OUTPUT OF OBTAINED MASK ==========================
   55 CALL PRIMSK(W,LW,LW,'OBTAINED MASK',13)
      PRINT*,' '
      PRINT*,'O = DESIRED OUTPUT IMAGE, B = INPUT IMAGE'
      REB=(B2+O2-2*BO)/O2
      REA=(O2-CRC)/O2
      WRITE(6,550) OA,O2,BO,BA,B2,CRC,REB,REA
  550 FORMAT(1H ,'O-Ave.L. =',F10.3,'  O2(Power) =',F15.3/
     &' BO(Co-Power) =',F15.3/
     &' B-Ave.L. =',F10.3,'  B2(Power) =',F15.3/
     &' CinvRC =',F15.3 /
     &' R-MSE/B-O/before (B2+O2-2*BO)/O2 = ',F15.3 /
     &' R-LMSE/RB-O/after (O2-CinvRC)/O2 = ',F15.3/)
      IF(ITERM.EQ.1) CALL TINPUT(ICH)
      GOTO 5
```

```
C====== FILTERING (APPLICATION OF W TO FF/G) ==========
   56 WRITE(6,560)
  560 FORMAT(1H ,'*** FILTERING (APPLICATION OF FILTER W TO G/FF)'/
     &' RESULT IS STORED INTO F ****'/
     &' SELECT-'/
     &' 1= TO G'/
     &' 2= TO FF'/)
      READ(5,*) ISW
      MM=M-2*LE
      NN=N-2*LE
      IF(ISW.EQ.2) GOTO 57
C-------------------------------------
      CALL LFTP(G,M,N,W,LE,LD,F,MM,NN)
      GOTO 58
C-------------------------------------
   57 CALL LFTP(FF,M,N,W,LE,LD,F,MM,NN)
C-------------------------------------
   58 NO=IUN*1000+LE
      CALL NORPKM(F,MM,NN,H,NL,NO)
      GOTO 111

C *** ADD NOISE ***********************************************************
    6 CALL CLEAR
      WRITE(6,600)
  600 FORMAT(1H ,'*** ADD NOISE ***'//
     &' 1= TO ORIGINAL IMAGE G'/
     &' 2= TO BUFFER (BLURRED) IMAGE FF'//)
      READ(5,*) ISW
      PRINT*,'INPUT SIGMA'
      PRINT*,' '
      READ(5,*) SIGMA
      PRINT*,'INPUT ANY LARGE ODD INTEGER'
      PRINT*,' '
      READ(5,*) ITN
      IF(ISW.NE.1) GOTO 60
C-------------------------------------
      CALL ANOISE(ITN,0.0,SIGMA,G,MN)
      GOTO 10
C-------------------------------------
   60 CALL ANOISE(ITN,0.0,SIGMA,FF,MN)
      GOTO 51

C***** FILTERINGS ***********************************
   80 CALL CLEAR
      PRINT*,'1=ENHANCEMENT'
```

```
      PRINT*,'2=P-MEANS'
      PRINT*,'3=DIRECTIONAL SMOOTHING'
      READ(5,*) ISW
      PRINT*,'MASK SIZE L?'
      READ(5,*) L
      MM=M-2*L
      NN=N-2*L
      IF(ISW.LT.1.OR.ISW.GT.3) GOTO 80
      GOTO (81,82,83),ISW
C-----------------------------------------
   81 CALL ENHFLT(G,M,N,L,F,MM,NN)
      GOTO 89
C-----------------------------------------
   82 PRINT*,'INPUT P'
      READ(5,*) P
      CALL PMFLT(G,M,N,L,P,F,MM,NN)
      GOTO 89
C-----------------------------------------------------
   83 CALL DSMOOTH(G,M,N,L,F,MM,NN,EV,2*L+1,R,4*L+1)
      GOTO 89
C-----------------------------------------------------
   89 NID=IUN*10000+(80+L)*100
      CALL NORPKM(F,MM,NN,H,NL,NID)
      GOTO 111

C********** SVD ****************************************
   85 CALL CLEAR
      PRINT*,'PARTITION MATRIX SIZE MP, NP?'
      READ(5,*) MP,NP
      MM=MP*(M/MP)
      NN=NP*(N/NP)
      PRINT*,'UP TO WHICH ORDER K < MIN(MP,NP)?'
      READ(5,*) K
C---------------------------
      DO 86 II=1,M/MP
      DO 86 JJ=1,N/NP
      I1=(II-1)*MP+1
      J1=(JJ-1)*NP+1
      CALL IMBED(G,M,N,I1,MP,J1,NP,FF,MP,NP,1,1 )
      CALL ANORM(FF,MP*NP,V)
      IF(V.LT.0.00001) GOTO 87
      CALL SVDNO2(FF,MP,NP,EV,R,W,K,1.0E-7)
      DO 88 I=1,K
      DO 88 J=1,NP
       JI=(I-1)*NP+J
```

```
   88  R(JI)=W(I)*R(JI)

       CALL MATMLT(3,EV,MP,K,R,NP,K,FF,MP,NP)
   87  CALL IMBED(FF,MP,NP,1,MP,1,NP,F,MM,NN,I1,J1)
   86  CONTINUE
       GOTO 111

C********** IMAGE DISPLAY ************************************
   90  PRINT*,'1= ORIGINAL G'
       PRINT*,'2= RESULTANT F'
       PRINT*,'3= BUFFER FF'
       PRINT*,'4= MASK P'
       PRINT*,'5= FILTER W'
       READ(5,*) ISW
       IF(ISW.LT.1.OR.ISW.GT.5) GOTO 90
       GOTO (15,89,51,92,93), ISW
   92  CALL NORPKM(P,M,N,H,NL,NID)
       GOTO 111
   93  CALL NORPKM(W,LW,LW,H,NL,NID)
       GOTO 111

C****** REPLACE IMAGES G, F, FF, P **********************
   70  CALL CLEAR
       PRINT*,'*** REPLACE IMAGE-'
       PRINT*,' 1= G WITH F (PACKED, M,N -> MM,NN)'
       PRINT*,' 2= G WITH FF'
       PRINT*,' 3= FF WITH G'
       PRINT*,' 4= FF WITH F (IMBED)'
       PRINT*,' 5= FF WITH P'
       PRINT*,' 6= EXIT TO MENUE'
       READ(5,*) ISW
       IF(ISW.LT.1.OR.ISW.GT.6) GOTO 70
       GOTO (71,72,73,74,75,111), ISW
C-------------- F => G ----------------------
   71  CALL SUBPACK(F,MM,NN,G,MN,MM,NN)
       M=MM
       N=NN
       GOTO 70
C-------------- FF => G ---------------------
   72  CALL MATMOV(G,M,N,FF)
       GOTO 70
C-------------- G => FF ---------------------
   73  CALL MATMOV(FF,M,N,G)
       GOTO 70
C-------------- F => FF ---------------------
```

```
   74 CALL IMBED(F,MM,NN,1,MM,1,NN,FF,M,N,L+1,L+1)
      GOTO 70
C-------------- P => FF ------------------------
   75 CALL MATMOV(FF,M,N,P)
      GOTO 70


C******** THRESHOLDING ******************
   97 CALL CLEAR
      PRINT*,'**** THRESHOLDING *****'
      PRINT*,'1= G TO FF'
      PRINT*,'2= F TO F'
      PRINT*,'3= FF TO FF'
      PRINT*,'4= EXIT TO MENU'
      READ(5,*) ISW
      GOTO( 98,99,91,111 ), ISW
   98 CALL THRESAN(G,M,N,NL,H,HB,FF,ITERM)
      GOTO 51
   99 CALL THRESAN(F,MM,NN,NL,H,HB,F,ITERM)
      GOTO 90
   91 CALL THRESAN(FF,M,N,NL,H,HB,FF,ITERM)
      GOTO 51



C********** HANDLING NORPAK DISPLAY FACILITIES ************
C============= INITIALIZE NORPAK ========================
   20 CONTINUE
      I=0
      PRINT*,' INITIALIZATION OF NORPAK'
      PRINT*,'1=ERASE ALL OLD MEMORY'
      PRINT*,'2=NOT ERASE OLD MEMORY'
      READ(5,*) ISW
      IF(ISW.EQ.1) I=1
      CALL ZINIT(I)
C------  LOAD CALIBRATE TABLES  ------------
      DO 200 K=1,256
      ILTP(K)=K-1
  200 CONTINUE

      CALL ZPSUDF
      DO 210 K=1,3
      CALL ZLLUT(K+3,ILTP)
      CALL ZCALBN(K)
  210 CONTINUE
```

```
      GOTO 111

C================ COLOUR CONTROL ====================
   25 PRINT*,'1= REAL-COLOUR (INITIAL DEFAULT)'
      PRINT*,'2= GRAY-LEVEL'
      PRINT*,'3= PSEUDO-COLOUR'
      PRINT*,'4= NEGATION (EXCEPT 3)'
      PRINT*,'5= EXIT (MENUE)'
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.5) GOTO 25
      GOTO(28,29,26,24,111),ISW

C-------- TURN GRAY LEVEL ON -----------------
   29 CONTINUE
      CALL ZPSUDF
      DO 291 K=1,3
      CALL ZLLUT(K+3,ILTP)
      CALL ZCALBN(K)
  291 CONTINUE

  292 PRINT*, 'BANK ? [1,2,3]  4= EXIT'
      READ(5,*) KBANK
      IF(KBANK.LT.1.OR.KBANK.GT.4) GOTO 292
      IF(KBANK.EQ.4) GOTO 25
      CALL ZCOLOR(KBANK,KBANK,KBANK)
      GOTO 292

C------------ TURN REAL COLOUR ON -----------------
   28 CONTINUE
      CALL ZPSUDF

      DO 281 K=1,3
      CALL ZLLUT(K+3,ILTP)
      CALL ZCALBN(K)
  281 CONTINUE

   27 PRINT*,'COLOUR ASSIGN -'
      PRINT*,' R TO BANK [1,2,3] 4=OFF'
      READ(5,*) IBR
      PRINT*,' G TO BANK [1,2,3] 4=OFF'
      READ(5,*) IBG
      PRINT*,' B TO BANK [1,2,3] 4=OFF'
      READ(5,*) IBB
      PRINT*,'R,  G,  B  =  BANK',IBR,IBG,IBB
      CALL ZCOLOR(IBR,IBG,IBB)
```

```
      PRINT*,'1= TRY AGAIN, 2= EXIT ?'
      READ(5,*) ISW
      IF(ISW.EQ.2) GOTO 25
      GOTO 27

C------------ PSEUDO COLOUR ------------------------------
   26 DO 261 K=1,3
      CALL ZCALBF(K)
  261 CONTINUE

  262 PRINT*,'CALIB.TABLE: 1= SMOOTH, 2= FINE ?'
      READ(5,*) ISW
      IF(ISW-2) 263, 264, 262

  264 DO 267 K=1,256
      K2=(K-1)/8
      IB=K-1-K2*8
      K3=K2/8
      IG=K2-K3*8
      ILTI(K)=85*K3
      ILTT(K)=36*IG
      ILTD(K)=36*IB
  267 CONTINUE
      GOTO 268

  263 DO 266 K=1,256
      IK=2*K-258
      IF(IK.LT.0) IK=0
      ILTI(K)=IK
      ILTD(257-K)=IK
      ILTT(K)=255-IABS(2*K-257)
  266 CONTINUE

  268 CALL ZLLUT(0,ILTI)
      CALL ZLLUT(1,ILTT)
      CALL ZLLUT(2,ILTD)

  265 PRINT*,'BANK ? [1,2,3]  4= EXIT'
      READ(5,*) KBANK
      IF(KBANK.LT.1.OR.KBANK.GT.4) GOTO 265
      IF(KBANK.EQ.4) GOTO 25
      CALL ZPSUDN(KBANK)
      GOTO 265

C------------ NEGATION -----------------------------------
```

```
   24 CONTINUE
      DO 240 K=1,256
      ILTP(K)= 255 - ILTP(K)
  240 CONTINUE
      DO 241 K=1,3
      CALL ZLLUT(K+3,ILTP)
  241 CONTINUE
      GOTO 25

C======== DATA NORMALIZATION ========
   21 CONTINUE
      WRITE(6,139)
  139 FORMAT(1H ,'DATA NORMALIZATION (INTO MEAN=0 & SD=1)'/
     &' 1= to G'/
     &' 2= to FF'/
     &' 3= EXIT TO MENUE'/)
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.3) GOTO 21
      IF(ISW.EQ.1) CALL NSTD(G,M,N,AVE,SD)
      IF(ISW.EQ.2) CALL NSTD(FF,M,N,AVE,SD)
      PRINT*,'AVE, SD =',AVE,SD
      GOTO 111

C====================== ERASING ============================
   22 CONTINUE
      PRINT*,'1= BANK1'
      PRINT*,'2= BANK2'
      PRINT*,'3= BANK3'
      PRINT*,'4= AREA'
      READ(5,*) ISW
      PRINT*,'ARE YOU SURE? 1=YES, 2=NO'
      READ(5,*) IYN
      IF(IYN.NE.1) GOTO 22
      IF(ISW.EQ.4) GOTO 23
      CALL ZERASE(ISW,255)
      GOTO 111
   23 PRINT*,'*** AREA ERASING MODE'
      PRINT*,'IDIM (ALONG ROW)?'
      READ(5,*) ID
      PRINT*,'JDIM (ALONG COLUMN)?'
      READ(5,*) JD
      CALL NORPK(P,ID,JD,0.0,0.0,0)
      GOTO 111

C****************** END ****************************************
```

```
    9 STOP
      END


      SUBROUTINE NORPK(F,MD,ND,A,B,IP)
C=============================================================
C     SUBROUTINE TO OUTPUT IMAGE TO NORPAK COLOUR DISPLAY
C     N.OTSU JUNE-8,'83
C-------------------------------------------------------------
CI     F(MD,ND)= OUTPUT IMAGE
CI     MD= NUMBER OF PIXELS IN ROW
CI     ND= NUMBER OF ROWS
CI     A = LOWER LIMIT OF RANGE
CI     B = UPPER LIMIT OF RANGE
CI     IP= SWITCH 1=POSI, -1=NEGA, 0=ERASE
C-------------------------------------------------------------
CW     I1= STARTING COLUMN ON NORPAK MAP IMAGE
CW     J1= STARTING ROW ON NORPAK MAP IMAGE
CW     KS= SCALE FACTOR FOR ENLARGEMENT OF F
CW     NB= THE NUMBER OF BANK TO LOAD THE IMAGE F (NB=1,2,3)
C=============================================================
      DIMENSION F(MD,ND)
      INTEGER*2 IBUFF(640)

C---------------< PARAMETERS SETTING >----------------------
  401    WRITE(5,400)
  400 FORMAT(' *** OUTPUT TO NORPK ***'//
     &' STORRING BANK [1,2,3]?, 4= QUIT (EXIT)')
      READ(5,*) NB
      IF(NB.LT.1.OR.NB.GT.4) GOTO 401
      IF(NB.EQ.4) GOTO 90

  101    WRITE(5,100)
  100 FORMAT(' STARTING COLUMN ON NORPAK [1-640]?')
      READ(5,*) I1
      IF(I1.LT.1.OR.I1.GT.640) GOTO 101

  201    WRITE(5,200)
  200 FORMAT(' STARTING ROW ON NORPAK [1-480]?')
      READ(5,*) J1
      IF(J1.LT.1.OR.J1.GT.480) GOTO 201

  301    WRITE(5,300)
  300 FORMAT(' SCALING FACTOR [1,2,...]?')
```

```
      READ(5,*) KS
      IF(KS.LT.1.OR.KS.GT.480) GOTO 301

  501 WRITE(6,500)
  500 FORMAT(' ARE YOU SURE ?'/ ' 1=YES, 2=NO')
      READ(5,*) ISW
      IF(ISW.EQ.2) GOTO 401

C---------< WRITE TO NORPAK >----------------
      CALL ZMODE(NB)
      DO 10 I=1,640
   10 IBUFF(I)=0

      NP=KS*MD
      NR=KS*ND
      IF(NP+I1.GT.641) NP=641-I1
      IF(NR+J1.GT.481) NR=481-J1
      IF(IP.EQ.0) GOTO 13

      EPS=1.E-10
      DEF=ABS(B-A)
      IF(DEF.LT.EPS) GOTO 11
      C=255/DEF
      D=-C*A
      IF(IP.EQ.1) GOTO 12
      C=-C
      D=-C*B
      GOTO 12

C----- THRESHOLDING ----------
   11 CONTINUE
      DO 20 J=1,NR
      JJ=(J-1)/KS+1

      DO 30 I=1,NP
      II=(I-1)/KS+1
      G=255.0
      IF(F(II,JJ).LT.A) G=0.0
      IF(IP.EQ.-1) G=255-G
      IBUFF(I)= G
   30 CONTINUE
      CALL ZWROW(J+J1-1,I1,NP,IBUFF)
   20 CONTINUE
      GOTO 401
```

```
C-------- RESCALING ------------
   12 CONTINUE
      DO 40 J=1,NR
      JJ=(J-1)/KS+1

      DO 50 I=1,NP
      II=(I-1)/KS+1
      G=C*F(II,JJ)+D
      G= AMAX1(G, 0.0)
      G= AMIN1(G, 255.0)
      IBUFF(I)= G
   50 CONTINUE
      CALL ZWROW(J+J1-1,I1,NP,IBUFF)
   40 CONTINUE
      GOTO 401

C-------- ERAZING ------------
   13 CONTINUE
      DO 60 J=1,NR
      CALL ZWROW(J+J1-1,I1,NP,IBUFF)
   60 CONTINUE
      GOTO 401

C----------------------------
   90 RETURN
      END


      SUBROUTINE NORPKM(G,M,N,H,L,NID)
C=================================================================
C   OPTIONAL DISPLAY OF IMAGE TO NORPAK
C   USING SUB.NORPK /N.OTSU, FEB.16,1983
C   * Intended not to alter G and NID
C-----------------------------------------------------------------
CI  G = given real image in any scale
CI  M= row-size (number of columns)
CI  N= number of rows
CO  H = histogram in L levels
CI  L= number of gray-levels for output, [EPS,L-EPS] (L <= 300)
C      all output image are stretched to this ful-range
CI  NID = image identification number
C=================================================================
      DIMENSION G(M,N),H(L),HB(300)

   11 WRITE(6,100)
```

```
  100 FORMAT(1H ,'SELECT -'/
     &,' 1= output as is (in ful-range)'/
     &,' 2= gray-scale inversion'/
     &,' 3= clipping negatives'/
     &,' 4= clipping positives & inversion'/
     &,' 5= clipping at upper and lower percentiles'/
     &,' 6= thresholding'/
     &,' 7= quit')
      READ(5,*) ISW
      IF(ISW.LT.1.OR.ISW.GT.7) GOTO 11
      IF(ISW.EQ.7) RETURN

C---------< ISSUE OF ID >-------------------
      WRITE(5,500) NID, ISW
  500 FORMAT(' ID = ',I8,'   OUT = ',I2)
      IF(ISW.GE.5) GOTO 7

C-----< MIN-MAX DETECTION >-----------------
      VMAX=-1.E10
      VMIN=1.E10
      DO 10 J=1,N
      DO 10 I=1,M
      GV=G(I,J)
      IF(GV.GT.VMAX) VMAX=GV
      IF(GV.LT.VMIN) VMIN=GV
   10 CONTINUE
      WRITE(6,400) VMIN, VMAX
  400 FORMAT(1H ,'VMIN, VMAX = ',2F10.3)

      GOTO (1,2,3,4), ISW

    1 CALL NORPK(G,M,N,VMIN,VMAX,1)
      RETURN
    2 CALL NORPK(G,M,N,VMIN,VMAX,-1)
      RETURN
    3 CALL NORPK(G,M,N,0.0,VMAX,1)
      RETURN
    4 CALL NORPK(G,M,N,VMIN,0.0,-1)
      RETURN

C=======================================
    7 CALL NRANGE(G,M,N,L)
      CALL NHIST(G,M,N,L,H,HMAX)
      IF(ISW-5) 5,5,6
```

```
C---------- PERCENTILE CUT ---------------
    5 PRINT*,'LOWER CUTOFF PERCENTILE (% >=0 )?'
      READ(5,*) PL
      PRINT*,'UPPER CUTOFF PERCENTILE (% >=0 )?'
      READ(5,*) PU
      PL=PL/100.0
      PU=PU/100.0
      GPL=0.0
      GPU=L

      SUML=0.0
      DO 35 I=1,L
      SUML=SUML+H(I)
      IF(SUML.GT.PL) GOTO 36
   35 CONTINUE
   36 GPL=I-1

      SUMU=0.0
      DO 37 I=1,L
      SUMU=SUMU+H(L+1-I)
      IF(SUMU.GT.PU) GOTO 38
   37 CONTINUE
   38 GPU=L+1-I

      CALL NORPK(G,M,N,GPL,GPU,1)
      RETURN

C-------- THRESHOLDING -----------------
    6 LST=550/L
      CALL HISTOP(H,L,450,350,LST,25,HMAX/10.0,10,0)

      WRITE(6,220) NID,L,L
  220 FORMAT(1H ,'*** THRESHOLDING FOR ID=',I8,' ***'//
     & ' GRAY LEVELS = [1,',I4,']'/
     & ' O( ) FOR [1, LTH], 1(*) FOR [LTH+1,',I4,']'//)
      PRINT*,'1=AUTO, 2=MANU'
      PRINT*,' '
      READ(5,*) ISW
      IF(ISW-1) 26,26, 23

   23 PRINT*,'INPUT LTH '
      PRINT*,' '
      READ(5,*) LTH
      GOTO 27
```

```
 26 CALL THRESA(H,L,AVE,VAR,LTH,P0,P1,U0,U1,SEP,HB)
    WRITE(6,200) AVE,VAR,LTH,P0,P1,U0,U1,SEP
200 FORMAT(1H0, '*** THRESA ***'//
   & ' TOTAL AVE. OF LEVEL = ',F7.1/
   & ' TOTAL VAR. OF LEVEL = ',F7.1//
   & ' OPTIMAL THRESHOLD LEVEL OBTAINED LTH = ',I4/
   & ' CLASS PROB. P0 = ',F5.1/
   & '              P1 = ',F5.1//
   & ' " AVE. LEV. U0 = ',F5.1/
   & '              U1 = ',F5.1//
   & ' SEP. MEASURE   = ',F5.3//)
    CALL HISTOP(HB,L,450,100,LST,15,0.1,10.0)

 27 CALL MOVABS(450+LST*LTH,300)
    CALL DRWREL(0,40)
    IF(ITERM.EQ. 1) CALL TINPUT(ICH)
    CALL CLEAR
    TH=LTH
    CALL NORPK(G,M,N,TH,TH,1)

    RETURN
    END
```